Amazon Virtual Private Cloud

User Guide

API Version 2015-04-15

Amazon Virtual Private Cloud: User Guide

Copyright © 2015 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Table of Contents

What is Amazon VPC? ... 1
- Amazon VPC Concepts ... 1
  - VPCs and Subnets ... 1
  - Supported Platforms ... 2
  - Default and Nondefault VPCs ... 2
  - Accessing the Internet ... 2
  - Accessing a Corporate or Home Network ... 4
- How to Get Started with Amazon VPC ... 5
- Services that Support Amazon VPC ... 6
- Accessing Amazon VPC ... 6
- Pricing for Amazon VPC ... 7
- Amazon VPC Limits ... 7

Getting Started ... 8
- Step 1: Create the VPC ... 9
  - Viewing Information About Your VPC ... 10
- Step 2: Create a Security Group ... 11
  - Rules for the WebServerSG Security Group ... 11
  - Creating Your WebServerSG Security Group ... 12
- Step 3: Launch an Instance into Your VPC ... 13
- Step 4: Assign an Elastic IP Address to Your Instance ... 14
- Step 5: Clean Up ... 15

VPC Wizard Scenarios for Amazon VPC ... 17
- Scenario 1: VPC with a Single Public Subnet ... 17
  - Configuration for Scenario 1 ... 18
  - Basic Components for Scenario 1 ... 18
  - Routing for Scenario 1 ... 19
  - Security for Scenario 1 ... 19
  - Implementing Scenario 1 ... 20
- Scenario 2: VPC with Public and Private Subnets (NAT) ... 22
  - Configuration for Scenario 2 ... 23
  - Basic Components for Scenario 2 ... 23
  - Routing for Scenario 2 ... 24
  - Security for Scenario 2 ... 25
  - Implementing Scenario 2 ... 28
- Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access ... 32
  - Configuration for Scenario 3 ... 32
  - Basic Configuration for Scenario 3 ... 33
  - Routing for Scenario 3 ... 33
  - Security for Scenario 3 ... 35
  - Implementing Scenario 3 ... 37
- Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access ... 41
  - Configuration for Scenario 4 ... 41
  - Basic Components for Scenario 4 ... 42
  - Routing for Scenario 4 ... 43
  - Security for Scenario 4 ... 43
  - Implementing Scenario 4 ... 44

Your VPC and Subnets ... 47
- Your VPC ... 47
  - Your New VPC ... 47
  - VPC Sizing ... 48
  - Connections with Your Local Network and Other VPCs ... 49
  - Creating a VPC ... 49
  - Deleting Your VPC ... 50
- Subnets in Your VPC ... 50
  - Your VPC with Subnets ... 51

Subnet Sizing 52
Subnet Routing 53
Subnet Security 53
Adding a Subnet to Your VPC 54
Launching an Instance into Your Subnet 54
Deleting Your Subnet 54
CLI Overview 55
Your Default VPC and Subnets 57
Default VPC Basics 57
Availability 57
Components 58
Default Subnets 59
Detecting Your Supported Platforms and Whether You Have a Default VPC 59
Detecting Platform Support Using the Console 59
Detecting Platform Support Using the Command Line 60
Launching an EC2 Instance into Your Default VPC 60
Launching an EC2 Instance Using the Console 60
Launching an EC2 Instance Using the Command Line 60
Deleting Your Default VPC 61
Security in Your VPC 62
Comparison of Security Groups and Network ACLs 62
Security Groups 64
Security Group Basics 64
Default Security Group for Your VPC 64
Security Group Rules 65
Differences Between Security Groups for EC2-Classic and EC2-VPC 66
Working with Security Groups 67
API and CLI Overview 69
Network ACLs 70
Network ACL Basics 71
Network ACL Rules 71
Default Network ACL 71
Example Custom Network ACL 72
Ephemeral Ports 74
Working with Network ACLs 74
API and Command Overview 131
Recommended Network ACL Rules for Your VPC 78
Recommended Rules for Scenario 1 79
Recommended Rules for Scenario 2 80
Recommended Rules for Scenario 3 83
Recommended Rules for Scenario 4 86
Controlling Access 87
Example Policies for a CLI or SDK 87
Example Policies for the Console 97
VPC Flow Logs 106
Flow Logs Basics 107
Flow Log Limitations 107
Flow Log Records 108
IAM Roles for Flow Logs 109
Working With Flow Logs 110
Troubleshooting 112
API and CLI Overview 113
Examples: Flow Log Records 114
Example: Creating a CloudWatch Metric Filter and Alarm for a Flow Log 115
Networking in Your VPC 116
IP Addressing 116
Public and Private IP Addresses 116
Modifying Your Subnet's Public IP Addressing Behavior 117

Assigning a Public IP Address During Launch ... 118
Elastic IP Addresses ... 119
Network Interfaces ... 122
Route Tables ... 123
Route Table Basics ... 123
Main Route Tables ... 123
Custom Route Tables ... 124
Route Table Association ... 125
Route Tables for VPC Peering Connections ... 126
Route Tables for ClassicLink ... 127
Route Tables for VPC Endpoints ... 127
Working with Route Tables ... 127
API and Command Overview ... 131
Internet Gateways ... 133
Creating a VPC with an Internet Gateway ... 134
NAT Instances ... 138
NAT Instance Basics ... 139
Setting up the NAT Instance ... 139
Creating the NATSG Security Group ... 141
Disabling Source/Destination Checks ... 142
Updating the Main Route Table ... 143
Testing Your NAT Instance Configuration ... 143
DHCP Options Sets ... 145
Overview of DHCP Options Sets ... 145
Amazon DNS Server ... 146
Changing DHCP Options ... 147
Working with DHCP Options Sets ... 147
API and Command Overview ... 149
DNS ... 149
Viewing DNS Hostnames for Your EC2 Instance ... 150
Updating DNS Support for Your VPC ... 151
Using Private Hosted Zones ... 152
VPC Peering ... 152
VPC Peering Basics ... 153
Working with VPC Peering Connections ... 155
API and CLI Overview ... 159
Controlling Access to VPC Peering Connections ... 160
VPC Endpoints ... 160
Endpoint Basics ... 161
Controlling the Use of Endpoints ... 165
Controlling Access to Services ... 165
Working with Endpoints ... 168
API and CLI Overview ... 170
Adding a Hardware Virtual Private Gateway to Your VPC ... 172
Components of Your VPN ... 173
Virtual Private Gateway ... 173
Customer Gateway ... 173
VPN Configuration Examples ... 173
Single VPN Connection ... 174
Multiple VPN connections ... 174
VPN Routing Options ... 174
What You Need for a VPN Connection ... 175
Configuring Two VPN Tunnels for Your VPN Connection ... 176
Using Redundant VPN Connections to Provide Failover ... 177
Setting Up the VPN Connection ... 177
Create a Customer Gateway ... 178
Create a Virtual Private Gateway ... 178
Enable Route Propagation in Your Route Table ... 178

Update Your Security Group to Enable Inbound SSH, RDP and ICMP Access 179
Create a VPN Connection and Configure the Customer Gateway 179
Launch an Instance Into Your Subnet 180
Testing the End-to-End Connectivity of Your Instance 180
Replacing Compromised Credentials 181
Editing Static Routes for a VPN Connection 181
Deleting a VPN Connection 182
API and CLI Overview 182
Providing Secure Communication Between Sites Using VPN CloudHub 185
Dedicated Instances 188
Dedicated Instance Basics 188
Dedicated Instances Limitations 189
Amazon EBS with Dedicated Instances 189
Reserved Instances with Dedicated Tenancy 189
Auto Scaling of Dedicated Instances 189
Pricing for Dedicated Instances 189
Working with Dedicated Instances 189
Creating a VPC with an Instance Tenancy of Dedicated 190
Launching Dedicated Instances into a VPC 190
Displaying Tenancy Information 190
API and Command Overview 191
ClassicLink 193
Amazon VPC Limits 194
Document History 197
AWS Glossary 200

What is Amazon VPC?

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

For more information about the benefits of using a VPC, see Benefits of Using a VPC in the *Amazon EC2 User Guide for Linux Instances*.

Topics

- Amazon VPC Concepts (p. 1)
- How to Get Started with Amazon VPC (p. 5)
- Services that Support Amazon VPC (p. 6)
- Accessing Amazon VPC (p. 6)
- Pricing for Amazon VPC (p. 7)
- Amazon VPC Limits (p. 7)

Amazon VPC Concepts

As you get started with Amazon VPC, you should understand the key concepts of this virtual network, and how it is similar to or different from your own networks. This section provides a brief description of the key concepts for Amazon VPC.

Amazon VPC is the networking layer for Amazon EC2. If you're new to Amazon EC2, see What is Amazon EC2? in the *Amazon EC2 User Guide for Linux Instances* to get a brief overview.

VPCs and Subnets

A *virtual private cloud* (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can configure your VPC; you can select its IP address range, create subnets, and configure route tables, network gateways, and security settings.

A *subnet* is a range of IP addresses in your VPC. You can launch AWS resources into a subnet that you select. Use a public subnet for resources that must be connected to the Internet, and a private subnet

for resources that won't be connected to the Internet. For more information about public and private subnets, see Your VPC with Subnets (p. 51).

To protect the AWS resources in each subnet, you can use multiple layers of security, including security groups and network access control lists (ACL). For more information, see Security in Your VPC (p. 62).

Supported Platforms

The original release of Amazon EC2 supported a single, flat network that's shared with other customers called the EC2-Classic platform. Older AWS accounts still support this platform, and can launch instances into either EC2-Classic or a VPC. Accounts created after 2013-12-04 support EC2-VPC only. For more information, see Detecting Your Supported Platforms and Whether You Have a Default VPC (p. 59).

Default and Nondefault VPCs

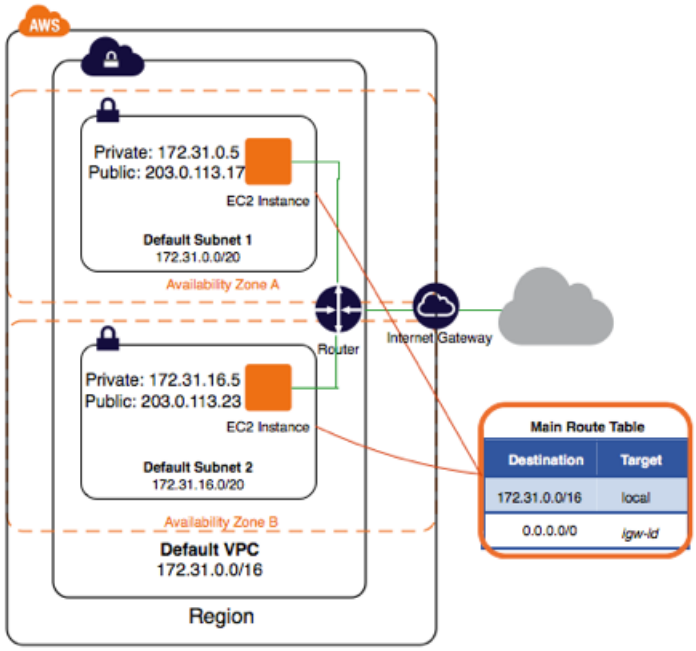
If your account supports the EC2-VPC platform only, it comes with a *default VPC* that has a *default subnet* in each Availability Zone. A default VPC has the benefits of the advanced features provided by EC2-VPC, and is ready for you to use. If you have a default VPC and don't specify a subnet when you launch an instance, the instance is launched into your default VPC. You can launch instances into your default VPC without needing to know anything about Amazon VPC. For more information, see Your Default VPC and Subnets (p. 57) and Launching an EC2 Instance into Your Default VPC (p. 60).

Regardless of which platforms your account supports, you can create your own VPC, and configure it as you need. This is known as a *nondefault VPC*. Subnets that you create in your nondefault VPC and additional subnets that you create in your default VPC are called *nondefault subnets*.

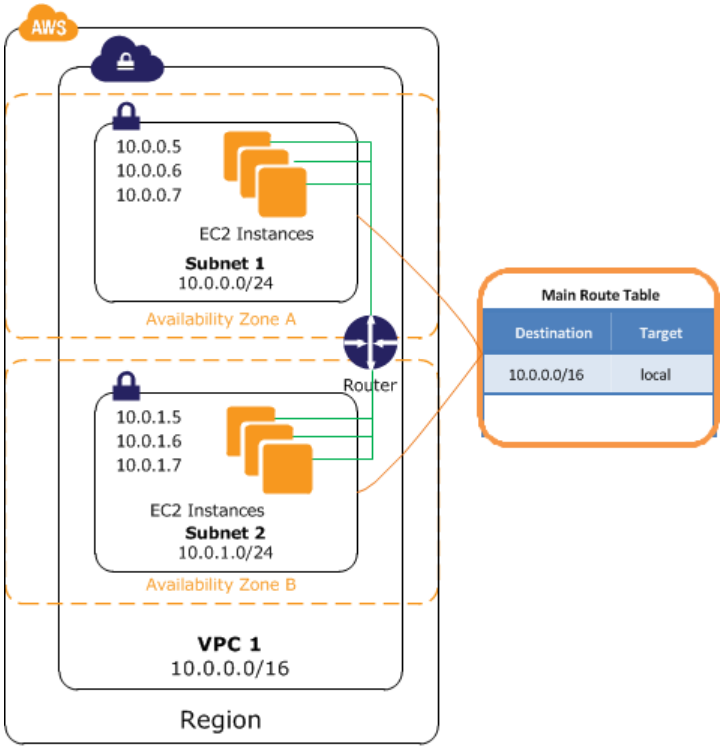
Accessing the Internet

You control how the instances that you launch into a VPC access resources outside the VPC.

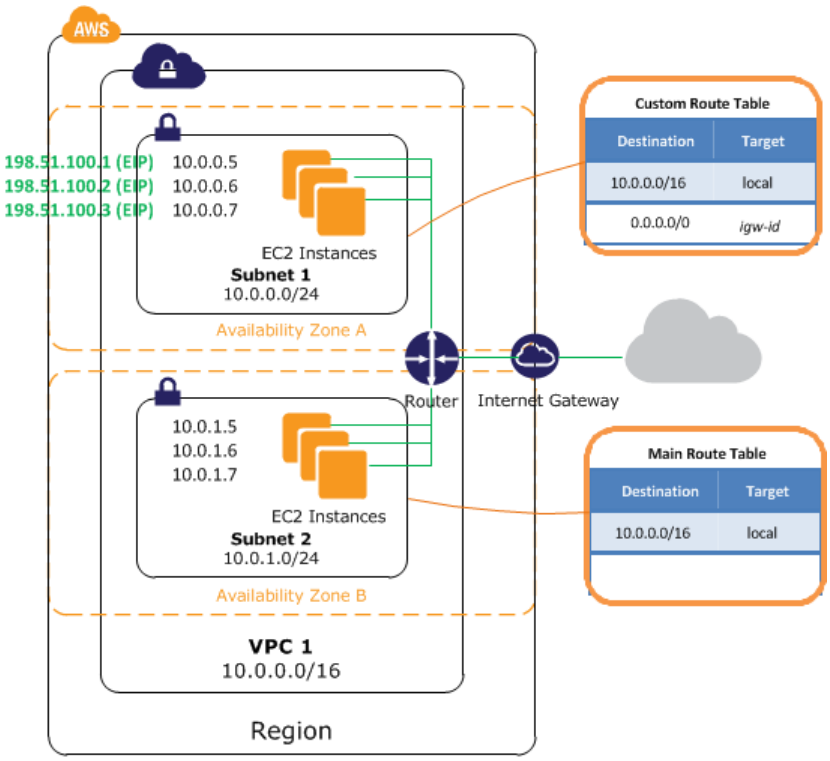
Your default VPC includes an Internet gateway, and each default subnet is a public subnet. Each instance that you launch into a default subnet has a private IP address and a public IP address. These instances can communicate with the Internet through the Internet gateway. An Internet gateway enables your instances to connect to the Internet through the Amazon EC2 network edge.



By default, each instance that you launch into a nondefault subnet has a private IP address, but no public IP address, unless you specifically assign one at launch, or you modify the subnet's public IP address attribute. These instances can communicate with each other, but can't access the Internet.



You can enable Internet access for an instance launched into a nondefault subnet by attaching an Internet gateway to its VPC (if its VPC is not a default VPC) and associating an Elastic IP address with the instance.



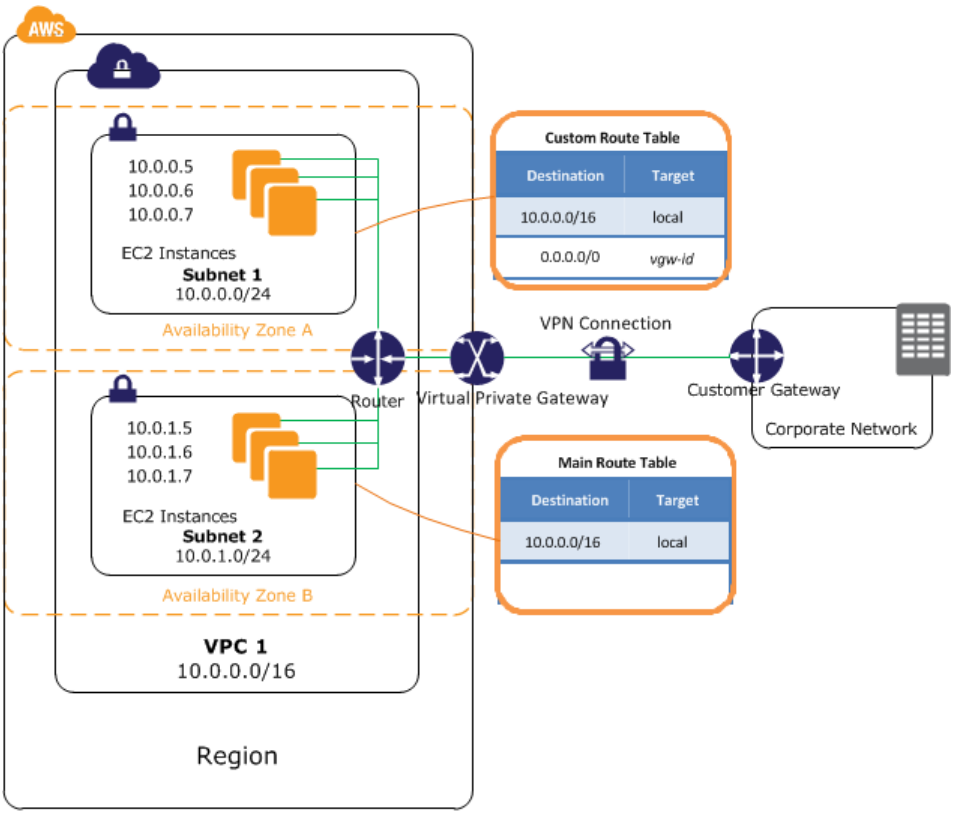
Alternatively, to allow an instance in your VPC to initiate outbound connections to the Internet but prevent unsolicited inbound connections from the Internet, you can use a network address translation (NAT) instance. NAT maps multiple private IP addresses to a single public IP address. A NAT instance has an Elastic IP address and is connected to the Internet through an Internet gateway. You can connect an instance in a private subnet to the Internet through the NAT instance, which routes traffic from the instance to the Internet gateway, and routes any responses to the instance.

For more information about routing and NAT in your VPC, see Route Tables (p. 123) and NAT Instances (p. 138).

Accessing a Corporate or Home Network

You can optionally connect your VPC to your own corporate data center using an IPsec hardware VPN connection, making the AWS cloud an extension of your data center.

A VPN connection consists of a virtual private gateway attached to your VPC and a customer gateway located in your data center. A virtual private gateway is the VPN concentrator on the Amazon side of the VPN connection. A customer gateway is a physical device or software appliance on your side of the VPN connection.



For more information, see Adding a Hardware Virtual Private Gateway to Your VPC (p. 172).

How to Get Started with Amazon VPC

To get a hands-on introduction to Amazon VPC, complete the exercise Getting Started with Amazon VPC (p. 8). The exercise will guide you through the steps to create a nondefault VPC with a public subnet, and to launch an instance into your subnet.

If you have a default VPC, and you want to get started launching instances into your VPC without performing any additional configuration on your VPC, see Launching an EC2 Instance into Your Default VPC (p. 60).

To learn about the basic scenarios for Amazon VPC, see VPC Wizard Scenarios for Amazon VPC (p. 17). You can configure your VPC and subnets in other ways to suit your needs. For more information about other scenarios, see Amazon Virtual Private Cloud Connectivity Options.

The following table lists related resources that you'll find useful as you work with this service.

Resource	Description
Amazon Virtual Private Cloud Connectivity Options	A whitepaper that provides an overview of the options for network connectivity.
Amazon VPC forum	A community-based forum for discussing technical questions related to Amazon VPC.

Resource	Description
AWS Developer Resources	A central starting point to find documentation, code samples, release notes, and other information to help you create innovative applications with AWS.
AWS Support Center	The home page for AWS Support.
Contact Us	A central contact point for inquiries concerning AWS billing, accounts, and events.

Services that Support Amazon VPC

To learn about using Amazon VPC with other AWS products, see the following documentation.

Service	Relevant Topic
AWS Data Pipeline	Launching Resources for Your Pipeline into a VPC
Amazon EC2	Amazon EC2 and Amazon VPC
Auto Scaling	Auto Scaling and Amazon VPC
Elastic Beanstalk	Using AWS Elastic Beanstalk with Amazon VPC
Elastic Load Balancing	Elastic Load Balancing and Amazon VPC
Amazon ElastiCache	Using ElastiCache with Amazon VPC
Amazon EMR	Select a Subnet for the Cluster
AWS OpsWorks	Running a Stack in a VPC
Amazon RDS	Amazon RDS and Amazon VPC
Amazon Redshift	Managing Clusters in a VPC
Amazon Route 53	Working with Private Hosted Zones

Accessing Amazon VPC

Amazon VPC provides a web-based user interface, the Amazon VPC console. If you've signed up for an AWS account, you can access the Amazon VPC console by signing into the AWS Management Console and selecting **VPC** from the console home page.

If you prefer to use a command line interface, you have several options:

AWS Command Line Interface (CLI)
: Provides commands for a broad set of AWS products, and is supported on Windows, Mac, and Linux/UNIX. To get started, see AWS Command Line Interface User Guide. For more information about the commands for Amazon VPC, see ec2.

Amazon EC2 Command Line Interface (CLI) Tools
: Provides commands for Amazon EC2, Amazon EBS, and Amazon VPC, and is supported on Windows, Mac, and Linux/UNIX. For more information about the commands, see Commands (CLI Tools) in the *Amazon EC2 Command Line Reference*.

AWS Tools for Windows PowerShell
: Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see AWS Tools for Windows PowerShell User Guide.

Amazon VPC provides a Query API. These requests are HTTP or HTTPS requests that use the HTTP verbs GET or POST and a Query parameter named `Action`. For more information about the API actions for Amazon VPC, see Actions in the *Amazon EC2 API Reference*.

If you prefer to build applications using language-specific APIs instead of submitting a request over HTTP or HTTPS, AWS provides libraries, sample code, tutorials, and other resources for software developers. These libraries provide basic functions that automatically take care of tasks such as cryptographically signing your requests, retrying requests, and handling error responses, so that it is easier for you to get started. For more information about downloading the AWS SDKs, see AWS SDKs and Tools.

Pricing for Amazon VPC

There's no additional charge for using Amazon VPC. You pay the standard rates for the instances and other Amazon EC2 features that you use. If you choose to create a hardware VPN connection, you pay for each hour that the VPN is connected to your VPC. For more information, see Amazon VPC Pricing and Amazon EC2 Pricing.

Amazon VPC Limits

There are limits to the number of Amazon VPC components that you can provision. You can request an increase in these limits. For more information about these limits, and how to request an increase, see Amazon VPC Limits (p. 194).

Getting Started with Amazon VPC

In this exercise, you'll create a VPC and subnet, and launch a public-facing instance into your subnet. Your instance will be able to communicate with the Internet, and you'll be able to access your instance from your local computer using SSH (if it's a Linux instance) or Remote Desktop (if it's a Windows instance). In your real world environment, you can use this scenario to create a public-facing web server; for example, to host a blog.

> **Note**
> This exercise is intended to help you set up your own nondefault VPC quickly. If you already have a default VPC and you want to get started launching instances into it (and not creating or configuring a new VPC), see Launching an EC2 Instance into Your Default VPC (p. 60).

To complete this exercise, you'll do the following:

- Create a nondefault VPC with a single public subnet. Subnets enable you to group instances based on your security and operational needs. A public subnet is a subnet that has access to the Internet through an Internet gateway.
- Create a security group for your instance that allows traffic only through specific ports.
- Launch an Amazon EC2 instance into your subnet.
- Associate an Elastic IP address with your instance. This allows your instance to access the Internet.

Before you can use Amazon VPC for the first time, you must sign up for Amazon Web Services (AWS). When you sign up, your AWS account is automatically signed up for all services in AWS, including Amazon VPC. If you haven't created an AWS account already, go to http://www.amazonaws.cn, and then choose **Create a Free Account**.

> **Note**
> This exercise assumes that your account supports the EC2-VPC platform only. If your account also supports the older EC2-Classic platform, you can still follow the steps in this exercise; however, you will not have a default VPC in your account to compare against your nondefault VPC. For more information, see Supported Platforms (p. 2).

Contents

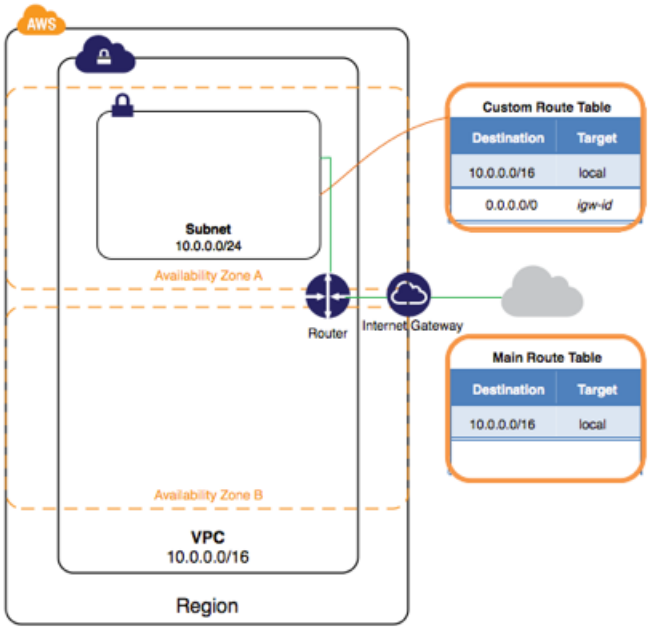
- Step 1: Create the VPC (p. 9)
- Step 2: Create a Security Group (p. 11)
- Step 3: Launch an Instance into Your VPC (p. 13)
- Step 4: Assign an Elastic IP Address to Your Instance (p. 14)
- Step 5: Clean Up (p. 15)

Step 1: Create the VPC

In this step, you'll use the Amazon VPC wizard in the Amazon VPC console to create a VPC. The wizard performs the following steps for you:

- Creates a VPC with a /16 CIDR block (a network with 65,536 private IP addresses). For more information about CIDR notation and the sizing of a VPC, see Your VPC (p. 47).
- Attaches an Internet gateway to the VPC. For more information about Internet gateways, see Internet Gateways (p. 133).
- Creates a size /24 subnet (a range of 256 private IP addresses) in the VPC.
- Creates a custom route table, and associates it with your subnet, so that traffic can flow between the subnet and the Internet gateway. For more information about route tables, see Route Tables (p. 123).

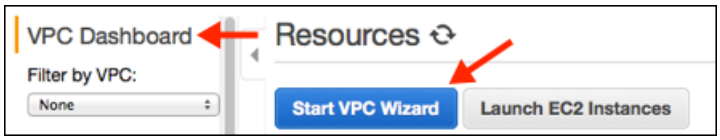
The following diagram represents the architecture of your VPC after you've completed this step.



Note
This exercise covers the first scenario in the VPC wizard. For more information about the other scenarios, see Scenarios for Amazon VPC.

To create a VPC using the Amazon VPC Wizard

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation bar, on the top-right, take note of the region in which you'll be creating the VPC. Ensure that you continue working in the same region for the rest of this exercise, as you cannot launch an instance into your VPC from a different region. For more information about regions, see Regions and Availability Zones.
3. On the Amazon VPC dashboard, choose **Start VPC Wizard**.

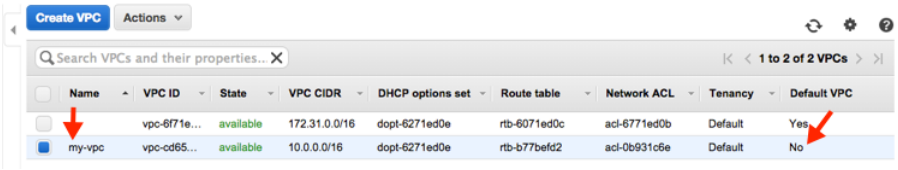


4. Choose the first option, **VPC with a Single Public Subnet**, and then choose **Select**.
5. On the configuration page, enter a name for your VPC in the **VPC name** field; for example, `my-vpc`, and enter a name for your subnet in the **Subnet name** field. This helps you to identify the VPC and subnet in the Amazon VPC console after you've created them. For this exercise, you can leave the rest of the configuration settings on the page, and choose **Create VPC**.

   (Optional) If you prefer, you can modify the configuration settings as follows, and then choose **Create VPC**.

   - The **IP CIDR block** displays the IP address range that you'll use for your VPC (`10.0.0.0/16`), and the **Public subnet** field displays the IP address range you'll use for the subnet (`10.0.0.0/24`). If you don't want to use the default CIDR ranges, you can specify your own. For more information, see VPC Sizing (p. 48) and Subnet Sizing (p. 52).
   - The **Availability Zone** list enables you to select the Availability Zone in which to create the subnet. You can leave **No Preference** to let AWS choose an Availability Zone for you. For more information, see Regions and Availability Zones.
   - In the **Add endpoints for S3 to your subnets** section, you can select a subnet in which to create a VPC endpoint to Amazon S3 in the same region. For more information, see VPC Endpoints (p. 160).
   - The **Enable DNS hostnames** option, when set to **Yes**, ensures that instances that are launched into your VPC receive a DNS hostname. For more information, see Using DNS with Your VPC (p. 149).
   - The **Hardware tenancy** option enables you to select whether instances launched into your VPC are run on shared or dedicated hardware. Selecting a dedicated tenancy incurs additional costs. For more information about hardware tenancy, see Dedicated Instances (p. 188).

6. A status window shows the work in progress. When the work completes, choose **OK** to close the status window.
7. The **Your VPCs** page displays your default VPC and the VPC that you just created. The VPC that you created is a nondefault VPC, therefore the **Default VPC** column displays **No**.



Viewing Information About Your VPC

After you've created the VPC, you can view information about the subnet, the Internet gateway, and the route tables. The VPC that you created has two route tables — a main route table that all VPCs have by default, and a custom route table that was created by the wizard. The custom route table is associated with your subnet, which means that the routes in that table determine how the traffic for the subnet flows. If you add a new subnet to your VPC, it uses the main route table by default.

To view information about your VPC

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Your VPCs**. Take note of the name and the ID of the VPC that you created (look in the **Name** and **VPC ID** columns). You will use this information to identify the components that are associated with your VPC.
3. In the navigation pane, choose **Subnets**. The console displays the subnet that was created when you created your VPC. You can identify the subnet by its name in **Name** column, or you can use the VPC information that you obtained in the previous step and look in the **VPC** column.
4. In the navigation pane, choose **Internet Gateways**. You can find the Internet gateway that's attached to your VPC by looking at the **VPC** column, which displays the ID and the name (if applicable) of the VPC.
5. In the navigation pane, choose **Route Tables**. There are two route tables associated with the VPC. Select the custom route table (the **Main** column displays **No**), and then choose the **Routes** tab to display the route information in the details pane:
   - The first row in the table is the local route, which enables instances within the VPC to communicate. This route is present in every route table by default, and you can't remove it.
   - The second row shows the route that the Amazon VPC wizard added to enable traffic destined for an IP address outside the VPC (`0.0.0.0/0`) to flow from the subnet to the Internet gateway.
6. Select the main route table. The main route table has a local route, but no other routes.

Step 2: Create a Security Group

A security group acts as a virtual firewall to control the traffic for its associated instances. To use a security group, you add the inbound rules to control incoming traffic to the instance, and outbound rules to control the outgoing traffic from your instance. To associate a security group with an instance, you specify the security group when you launch the instance. If you add and remove rules from the security group, we apply those changes to the instances associated with the security group automatically.

Your VPC comes with a *default security group*. Any instance not associated with another security group during launch is associated with the default security group. In this exercise, you'll create a new security group, `WebServerSG`, and specify this security group when you launch an instance into your VPC.

Topics

- Rules for the WebServerSG Security Group (p. 11)
- Creating Your WebServerSG Security Group (p. 12)

Rules for the WebServerSG Security Group

The following table describes the inbound and outbound rules for the `WebServerSG` security group. You'll add the inbound rules yourself. The outbound rule is a default rule that allows all outbound communication to anywhere — you do not need to add this rule yourself.

Inbound			
Source IP	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allows inbound HTTP access from anywhere.

0.0.0.0/0	TCP	443	Allows inbound HTTPS access from anywhere.
Public IP address range of your home network	TCP	22	Allows inbound SSH access from your home network to a Linux/UNIX instance.
Public IP address range of your home network	TCP	3389	Allows inbound RDP access from your home network to a Windows instance.
Outbound			
Destination IP	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	All	All	The default outbound rule that allows all outbound communication.

Creating Your WebServerSG Security Group

You can create your security group using the Amazon VPC console.

To create the WebServerSG security group and add rules

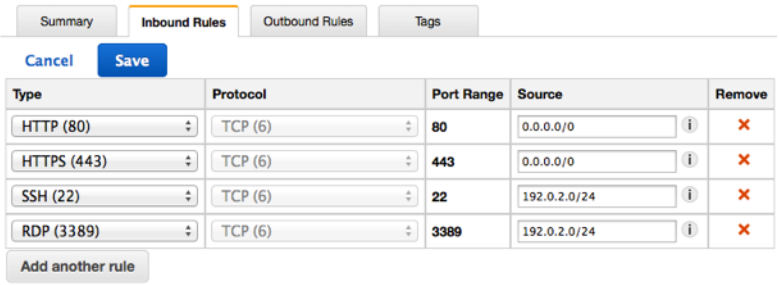
1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Security Groups**.
3. Choose **Create Security Group**.
4. In the **Group name** field, enter `WebServerSG` as the name of the security group, and provide a description. You can optionally use the **Name tag** field to create a tag for the security group with a key of `Name` and a value that you specify.
5. Select the ID of your VPC from the **VPC** menu, and then choose **Yes, Create**.
6. Select the `WebServerSG` security group that you just created (you can view its name in the **Group Name** column).
7. On the **Inbound Rules** tab, choose **Edit** and add rules for inbound traffic as follows, and then choose **Save** when you're done:

   a. Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.

   b. Choose **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.

   c. Choose **Add another rule**. If you're launching a Linux instance, select **SSH** from the **Type** list, or if you're launching a Windows instance, select **RDP** from the **Type** list. Enter your network's public IP address range in the **Source** field. If you don't know this address range, you can use `0.0.0.0/0` for this exercise.

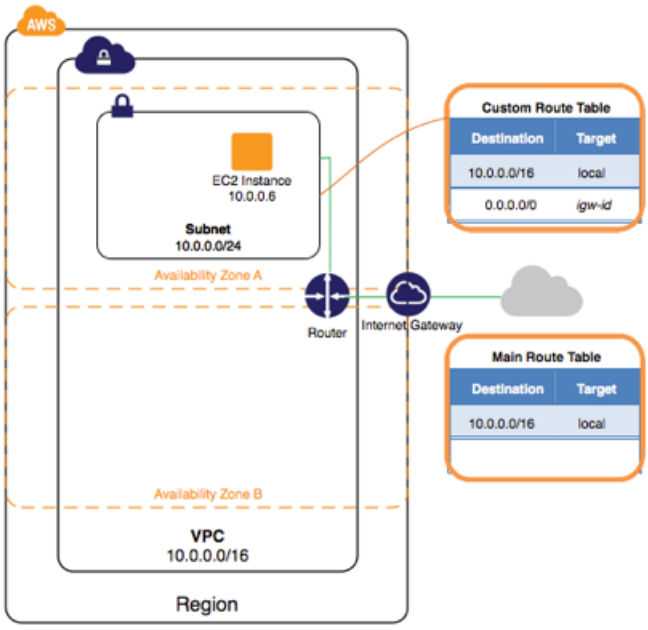
      Caution
      If you use `0.0.0.0/0`, you enable all IP addresses to access your instance using SSH or RDP. This is acceptable for the short exercise, but it's unsafe for production environments. In production, you'll authorize only a specific IP address or range of addresses to access your instance.



Step 3: Launch an Instance into Your VPC

When you launch an EC2 instance into a VPC, you must specify the subnet in which to launch the instance. In this case, you'll launch an instance into the public subnet of the VPC you created. You'll use the Amazon EC2 launch wizard in the Amazon EC2 console to launch your instance.

The following diagram represents the architecture of your VPC after you've completed this step.



To launch an EC2 instance into a VPC

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation bar, on the top-right, ensure that you select the same region in which you created your VPC and security group.
3. From the dashboard, choose **Launch Instance**.
4. On the first page of the wizard, choose the AMI that you want to use. For this exercise, we recommend that you choose an Amazon Linux AMI or a Windows AMI.
5. On the **Choose an Instance Type** page, you can select the hardware configuration and size of the instance to launch. By default, the wizard selects the first available instance type based on the AMI

you selected. You can leave the default selection, and then choose **Next: Configure Instance Details**.

6. On the **Configure Instance Details** page, select the VPC that you created from the **Network** list, and the subnet from the **Subnet** list. Leave the rest of the default settings, and go through the next pages of the wizard until you get to the **Tag Instance** page.
7. On the **Tag Instance** page, you can tag your instance with a `Name` tag; for example `Name=MyWebServer`. This helps you to identify your instance in the Amazon EC2 console after you've launched it. Choose **Next: Configure Security Group** when you are done.
8. On the **Configure Security Group** page, the wizard automatically defines the launch-wizard-*x* security group to allow you to connect to your instance. Instead, choose the **Select an existing security group** option, select the **WebServerSG** group that you created previously, and then choose **Review and Launch**.
9. On the **Review Instance Launch** page, check the details of your instance, and then choose **Launch**.
10. In the **Select an existing key pair or create a new key pair** dialog box, you can choose an existing key pair, or create a new one. If you create a new key pair, ensure that you download the file and store it in a secure location. You'll need the contents of the private key to connect to your instance after it's launched.

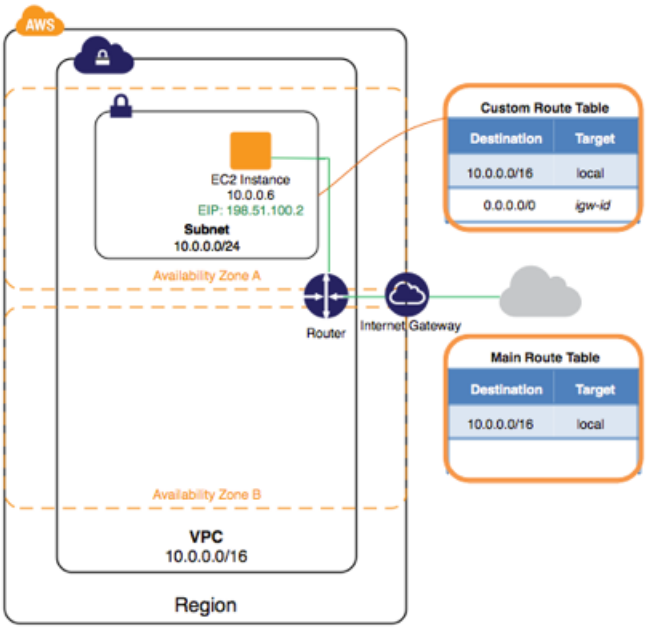
    To launch your instance, select the acknowledgment check box, and then choose **Launch Instances**.
11. On the confirmation page, choose **View Instances** to view your instance on the **Instances** page. Select your instance, and view its details in the **Description** tab. The **Private IPs** field displays the private IP address that's assigned to your instance from the range of IP addresses in your subnet.

For more information about the options available in the Amazon EC2 launch wizard, see Launching an Instance in the *Amazon EC2 User Guide for Linux Instances*.

Step 4: Assign an Elastic IP Address to Your Instance

In the previous step, you launched your instance into a public subnet — a subnet that has a route to an Internet gateway. However, the instance in your subnet also needs a public IP address to be able to communicate with the Internet. By default, an instance in a nondefault VPC is not assigned a public IP address. In this step, you'll allocate an Elastic IP address to your account, and then associate it with your instance. For more information about Elastic IP addresses, see Elastic IP Addresses (p. 119).

The following diagram represents the architecture of your VPC after you've completed this step.



To allocate and assign an Elastic IP address

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Elastic IPs**.
3. Choose **Allocate New Address**, and then **Yes, Allocate**.

   Note
   If your account supports EC2-Classic, first select **EC2-VPC** from the **Network platform** list.
4. Select the Elastic IP address from the list, and then choose **Associate Address**.
5. In the dialog box, choose **Instance** from the **Associate with** list, and then select your instance from the **Instance** list. Choose **Yes, Associate** when you're done.

Your instance is now accessible from the Internet. You can connect to your instance through its Elastic IP address using SSH or Remote Desktop from your home network. For more information about how to connect to a Linux instance, see Connecting to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For more information about how to connect to a Windows instance, see Connect to Your Windows Instance Using RDP in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

This completes the exercise; you can choose to continue using your instance in your VPC, or if you do not need the instance, you can terminate it and release its Elastic IP address to avoid incurring charges for them. You can also delete your VPC — note that you are not charged for the VPC and VPC components created in this exercise (such as the subnets and route tables).

Step 5: Clean Up

Before you can delete a VPC, you must terminate any instances that are running in the VPC. If you delete a VPC using the VPC console, it also deletes resources that are associated with the VPC, such as subnets, security groups, network ACLs, DHCP options sets, route tables, and Internet gateways.

To terminate your instance, release your Elastic IP address, and delete your VPC

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.

2. In the navigation pane, choose **Instances**.
3. Select your instance, choose **Actions**, then **Instance State**, and then select **Terminate**.
4. In the dialog box, expand the **Release attached Elastic IPs** section, and select the check box next to the Elastic IP address. Choose **Yes, Terminate**.
5. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
6. In the navigation pane, choose **Your VPCs**.
7. Select the VPC, choose **Actions**, and then choose **Delete VPC**.
8. When prompted for confirmation, choose **Yes, Delete**.

VPC Wizard Scenarios for Amazon VPC

This section describes how to use the VPC wizard to create basic scenarios for Amazon VPC. Each scenario includes the following information:

- A diagram showing the basic components
- Information about the VPC and subnets
- Information about the routing tables for the subnet
- Information about the recommended security group rules
- Step-by-step directions to implement the scenario

The following table describes the basic scenarios.

Scenario	Usage
Scenario 1: VPC with a Single Public Subnet (p. 17)	Run a single-tier, public-facing web application such as a blog or simple web site.
Scenario 2: VPC with Public and Private Subnets (NAT) (p. 22)	Run a public-facing web application, while still maintaining non-publicly accessible back-end servers in a second subnet.
Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 32)	Extend your data center into the cloud, and also directly access the Internet from your VPC.
Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access (p. 41)	Extend your data center into the cloud, and leverage Amazon's infrastructure without exposing your network to the Internet.

Scenario 1: VPC with a Single Public Subnet

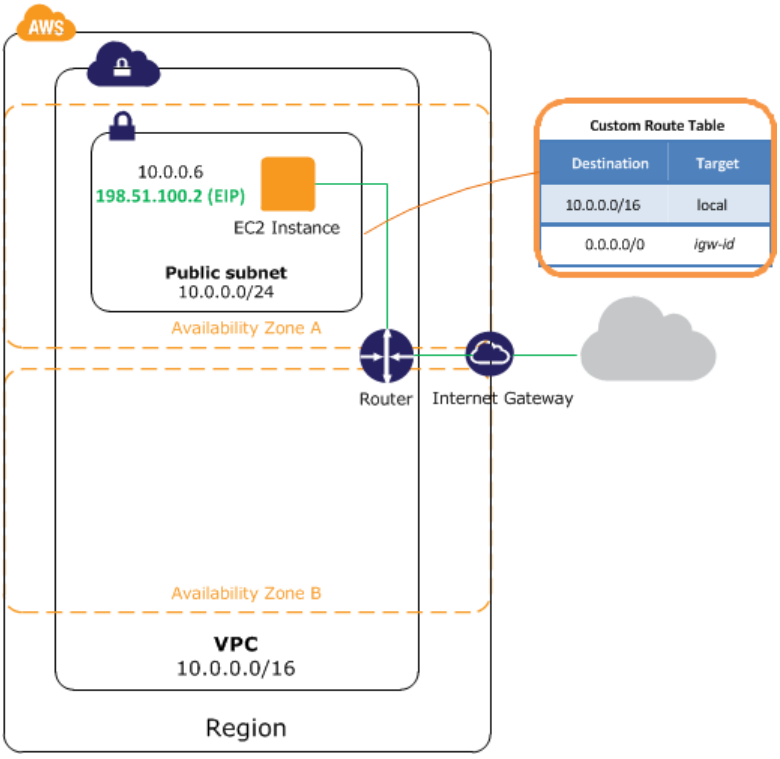
The configuration for this scenario includes a virtual private cloud (VPC) with a single public subnet, and an Internet gateway to enable communication over the Internet. We recommend this configuration if you need to run a single-tier, public-facing web application, such as a blog or a simple website.

Topics

- Configuration for Scenario 1 (p. 18)
- Basic Components for Scenario 1 (p. 18)
- Routing for Scenario 1 (p. 19)
- Security for Scenario 1 (p. 19)
- Implementing Scenario 1 (p. 20)

Configuration for Scenario 1

The following diagram shows the key components of the configuration for this scenario.



Note
If you completed the exercise Getting Started with Amazon VPC (p. 8), then you've already implemented this scenario using the VPC wizard in the Amazon VPC console.

Basic Components for Scenario 1

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- An Internet gateway. This connects the VPC to the Internet and to other AWS products.
- An instance with a private IP address in the subnet range (example: 10.0.0.6), which enables the instance to communicate with other instances in the VPC, and an Elastic IP address (example: 198.51.100.2), which enables the instance to be reached from the Internet.

- A route table entry that enables instances in the subnet to communicate with other instances in the VPC, and a route table entry that enables instances in the subnet to communicate directly over the Internet.

For more information about subnets, see Your VPC and Subnets (p. 47) and IP Addressing in Your VPC (p. 116). For more information about Internet gateways, see Internet Gateways (p. 133).

Tip
If you'd like instances in your VPC to communicate over the Internet without having to assign each instance an Elastic IP address, you can use a NAT instance. For more information about configuring a NAT instances, see Scenario 2: VPC with Public and Private Subnets (NAT) (p. 22) or NAT Instances (p. 138).

Routing for Scenario 1

Your VPC has an implied router (shown in the configuration diagram for this scenario.) For this scenario, the VPC wizard creates a route table that routes all traffic destined for an address outside the VPC to the Internet gateway, and associates this route table with the subnet. Otherwise, you'd need to create and associate the route table yourself.

The following table shows what the route table looks like for the example addresses used in the configuration diagram for this scenario. The first row shows the entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second row shows the entry for routing all other subnet traffic to the Internet gateway, which is specified using its AWS-assigned identifier.

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-*xxxxxxxx*

Security for Scenario 1

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see Security in Your VPC (p. 62).

For scenario 1, you'll use a security group but not network ACLs. If you'd like to use a network ACL, see Recommended Rules for Scenario 1 (p. 79).

Recommended Security Group Rules

Your VPC comes with a default security group whose initial settings allow all outbound traffic and allow all traffic between instances assigned to the security group. If you don't specify a security group when you launch an instance, the instance is automatically assigned to the default security group for the VPC. We could modify the rules for the default security group, but the rules that you need for your web servers might not work for other instances that you launch into the VPC. Therefore, we recommend that you create a security group to use with the web servers in your public subnet.

You'll create a security group named `WebServerSG`, modify the rules as needed, and then specify the security group when you launch instances into the VPC. By default, new security groups start with only an outbound rule that allows all traffic to leave the instances. You must add rules to enable any inbound traffic or to restrict the outbound traffic.

The following table describes the inbound and outbound rules for the WebServerSG group. If you want your web server to initiate outbound traffic, for example, to get software updates, you can leave the default outbound rule. If you do not want your web server to initiate outbound traffic, you can remove the default outbound rule.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
Public IP address range of your network	TCP	22	(Linux instances) Allow inbound SSH access from your network
Public IP address range of your network	TCP	3389	(Windows instances) Allow inbound RDP access from your network
Outbound (Optional)			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	All	All	Allow all outbound access to anywhere

Tip
You can also get the public IP address of your local computer using a service. To locate a service that provides your IP address, use the search phrase "what is my IP address". If you are connecting through an ISP or from behind a firewall without a static IP address, you need to find the range of IP addresses used by client computers.

The default security group for a VPC has rules that automatically allow assigned instances to communicate with each other. To allow that type of communication between the instances in your VPC, you must add a rule like the following to your security groups.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
The security group ID (sg-*xxxxxxxx*)	All	All	Allow inbound traffic from other instances assigned to this security group

Implementing Scenario 1

Use the following process to implement the scenario using the VPC wizard.

To implement scenario 1 using the VPC wizard

1. Set up the VPC, subnet, and Internet gateway:

a. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
b. In the navigation pane, click **VPC Dashboard**.
c. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
d. Select the first option, **VPC with a Single Public Subnet**, and then click **Select**.
e. The confirmation page shows the CIDR ranges and settings that you've chosen. Make any changes that you need, and then click **Create VPC** to create your VPC, subnet, Internet gateway, and route table.

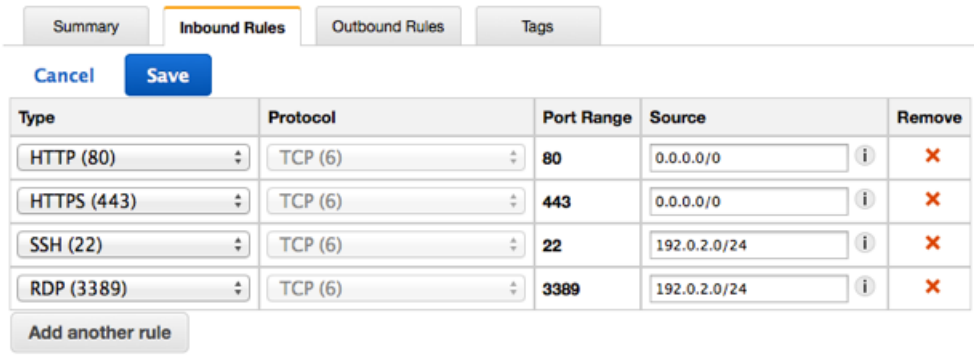
2. Create the WebServerSG security group and add rules:

a. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
b. In the navigation pane, click **Security Groups**.
c. Click the **Create Security Group** button.
d. Specify `WebServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** menu, and then click **Yes, Create**.
e. Select the WebServerSG security group that you just created. The details pane include a tab for information about the security group, plus tabs for working with its inbound rules and outbound rules.
f. On the **Inbound Rules** tab, click **Edit**, and then do the following:

- Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
- Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
- Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field. (If you don't know this address range, you can use `0.0.0.0/0` for testing purposes; in production, you'll authorize only a specific IP address or range of addresses to access your instance.)

  Tip
  If your company uses both Linux and Windows instances, you can add access for both SSH and RDP.

- Click **Save**.



g. (Optional) On the **Outbound Rules** tab, click **Edit**. Locate the default rule that enables all outbound traffic, click **Remove**, and then click **Save**.

3. Launch an instance into the VPC:

a. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
b. From the dashboard, click the **Launch Instance** button.
c. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
d. On the **Configure Instance Details** page, select the VPC that you created in step 1 from the **Network** list, and then specify a subnet.
e. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IP address. To be able to connect to your instance, you can assign a public IP address now, or allocate an Elastic IP address and assign it to your instance after it's launched. To assign a public IP address now, ensure that you select **Enable** from the **Auto-assign Public IP** list.

   Note
   You can only assign a public IP address to a single, new network interface with the device index of eth0. For more information, see Assigning a Public IP Address During Launch (p. 118).

f. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, select the **Select an existing security group** option, and select the **WebServerSG** security group that you created in step 2. Click **Review and Launch**.
g. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.

4. If you did not assign a public IP address to your instance as part of step 3, you will not be able to connect to it. Assign an Elastic IP address to the instance:

   a. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
   b. In the navigation pane, click **Elastic IPs**.
   c. Click the **Allocate New Address** button.
   d. Click **Yes, Allocate**.

      Note
      If your account supports EC2-Classic, first choose **EC2-VPC** from the **Network platform** list.

   e. Select the Elastic IP address from the list, and then click the **Associate Address** button.
   f. In the **Associate Address** dialog box, select the instance to associate the address with, and then click **Yes, Associate**.

You can now connect to your instances in the VPC. For information about how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Scenario 2: VPC with Public and Private Subnets (NAT)

The configuration for this scenario includes a virtual private cloud (VPC) with a public subnet and a private subnet. We recommend this scenario if you want to run a public-facing web application, while maintaining back-end servers that aren't publicly accessible. A common example is a multi-tier website, with the web

servers in a public subnet and the database servers in a private subnet. You can set up security and routing so that the web servers can communicate with the database servers.

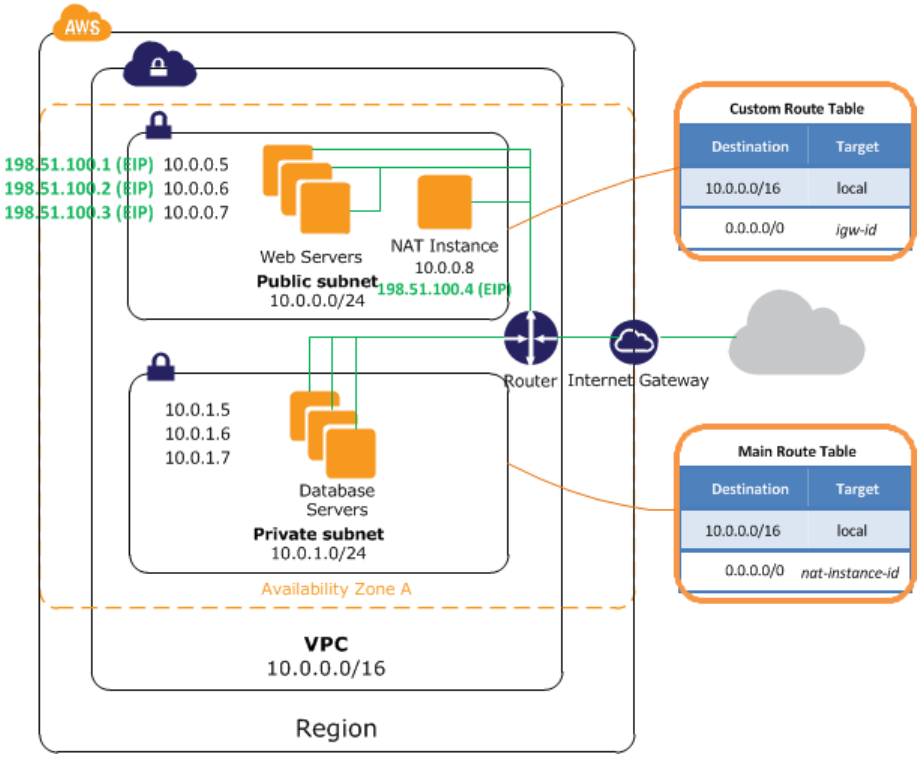
The instances in the public subnet can receive inbound traffic directly from the Internet, whereas the instances in the private subnet can't. The instances in the public subnet can send outbound traffic directly to the Internet, whereas the instances in the private subnet can't. Instead, the instances in the private subnet can access the Internet by using a network address translation (NAT) instance that you launch into the public subnet.

Topics

- Configuration for Scenario 2 (p. 23)
- Basic Components for Scenario 2 (p. 23)
- Routing for Scenario 2 (p. 24)
- Security for Scenario 2 (p. 25)
- Implementing Scenario 2 (p. 28)

Configuration for Scenario 2

The following diagram shows the key components of the configuration for this scenario.



Basic Components for Scenario 2

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A public subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- A private subnet of size /24 (example CIDR: 10.0.1.0/24). This provides 256 private IP addresses.

- An Internet gateway. This connects the VPC to the Internet and to other AWS products.
- Instances with private IP addresses in the subnet range (examples: 10.0.0.5, 10.0.1.5), which enables them to communicate with each other and other instances in the VPC. Instances in the public subnet also have Elastic IP addresses (example: 198.51.100.1), which enable them to be reached from the Internet. Instances in the private subnet are back-end servers that don't need to accept incoming traffic from the Internet; however, they can send requests to the Internet using the NAT instance (see the next bullet).
- A network address translation (NAT) instance with its own Elastic IP address. This enables instances in the private subnet to send requests to the Internet (for example, for software updates).
- A custom route table associated with the public subnet. This route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC, and an entry that enables instances in the subnet to communicate directly with the Internet.
- The main route table associated with the private subnet. The route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC, and an entry that enables instances in the subnet to communicate with the Internet through the NAT instance.

For more information about subnets, see Your VPC and Subnets (p. 47) and IP Addressing in Your VPC (p. 116). For more information about Internet gateways, see Internet Gateways (p. 133). For more information about NAT, see NAT Instances (p. 138).

> **Tip**
> To help manage the instances in the private subnet, you can set up bastion servers in the public subnet to act as proxies. For example, you can set up SSH port forwarders or RDP gateways in the public subnet to proxy the traffic going to your database servers from your own network.

Routing for Scenario 2

Your VPC has an implied router (shown in the configuration diagram for this scenario). For this scenario, the VPC wizard updates the main route table used with the private subnet, and creates a custom route table and associates it with the public subnet. Otherwise, you'd need to create and associate the route tables yourself.

In this scenario, all traffic from each subnet that is bound for AWS (for example, to the Amazon EC2 or Amazon S3 endpoints) goes over the Internet gateway. The database servers in the private subnet can't receive traffic from the Internet directly because they don't have Elastic IP addresses. However, the database servers can send and receive Internet traffic through the NAT instance in the public subnet.

Any additional subnets that you create use the main route table by default, which means that they are private subnets by default. If you'd like to make a subnet public, you can always change the route table that it's associated with.

The following tables describe the route tables for this scenario.

Main Route Table

The first row describes the entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second row describes the entry that sends all other subnet traffic to the NAT instance, which is specified using its AWS-assigned identifiers (for example, network interface `eni-1a2b3c4d` and instance `i-1a2b3c4d`).

Destination	Target
`10.0.0.0/16`	local
`0.0.0.0/0`	eni-*xxxxxxxx* / i-*xxxxxxxx*

Custom Route Table

The first row describes the entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic to the Internet over the Internet gateway, which is specified using its AWS-assigned identifier (for example, `igw-1a2b3d4d`).

Destination	Target
`10.0.0.0/16`	local
`0.0.0.0/0`	`igw-`*`xxxxxxxx`*

Security for Scenario 2

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see Security in Your VPC (p. 62).

For scenario 2, you'll use security groups but not network ACLs. If you'd like to use a network ACL, see Recommended Rules for Scenario 2 (p. 80).

Recommended Security Groups

Your VPC comes with a default security group whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances assigned to the group. If you don't specify a security group when you launch an instance, the instance is automatically assigned to this default security group.

For this scenario, we recommend that you create the following security groups instead of modifying the default security group:

- **WebServerSG—**For the web servers in the public subnet
- **NATSG—**For the NAT instance in the public subnet
- **DBServerSG—**For the database servers in the private subnet

The instances assigned to a security group can be in different subnets. However, in this scenario, each security group corresponds to the type of role an instance plays, and each role requires the instance to be in a particular subnet. Therefore, in this scenario, all instances assigned to a security group are in the same subnet.

The WebServerSG security group is the security group that you'll specify when you launch your web servers into your public subnet. The following table describes the recommended rules for this security group, which allow the web servers to receive Internet traffic, as well as SSH and RDP traffic from your network. The web servers can also initiate read and write requests to the database servers in the private subnet. Because the web server doesn't initiate outbound communication, we'll remove the default outbound rule.

Note
These recommendations include both SSH and RDP access, and both Microsoft SQL Server and MySQL access. For your situation, you might only need rules for Linux (SSH and MySQL) or Windows (RDP and Microsoft SQL Server).

WebServerSG: Recommended Rules

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
Your home network's public IP address range	TCP	22	Allow inbound SSH access to Linux instances from your home network (over the Internet gateway)
Your home network's public IP address range	TCP	3389	Allow inbound RDP access to Windows instances from your home network (over the Internet gateway)
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
The ID of your DBServerSG security group	TCP	1433	Allow outbound Microsoft SQL Server access to the database servers assigned to DBServerSG
The ID of your DBServerSG security group	TCP	3306	Allow outbound MySQL access to the database servers assigned to DBServerSG

The NATSG security group is the security group that you'll specify when you launch a NAT instance into your public subnet. The following table describes the recommended rules for this security group, which allow the NAT instance to receive Internet-bound traffic from instances in the private subnet, as well as SSH traffic from your network. The NAT instance can also send traffic to the Internet, so that instances in the private subnet can get software updates.

NATSG: Recommended Rules

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
10.0.1.0/24	TCP	80	Allow inbound HTTP traffic from database servers in the private subnet
10.0.1.0/24	TCP	443	Allow inbound HTTPS traffic from database servers in the private subnet
Your network's public IP address range	TCP	22	Allow inbound SSH access to the NAT instance from your network (over the Internet gateway)

Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow outbound HTTP access to the Internet (over the Internet gateway)
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (over the Internet gateway)

The DBServerSG security group is the security group that you'll specify when you launch your database servers into your private subnet. The following table describes the recommended rules for this security group, which allow read or write database requests from the web servers. The database servers can also initiate traffic bound for the Internet (your route table sends that traffic to the NAT instance, which then forwards it to the Internet over the Internet gateway).

DBServerSG: Recommended Rules

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
The ID of your WebServerSG security group	TCP	1433	Allow web servers assigned to WebServerSG Microsoft SQL Server access to database servers assigned to DBServerSG
The ID of your WebServerSG security group	TCP	3306	Allow web servers assigned to WebServerSG MySQL access to database servers assigned to DBServerSG
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow outbound HTTP access to the Internet (for example, for software updates)
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (for example, for software updates)

The default security group for a VPC has rules that automatically allow assigned instances to communicate with each other. To allow that type of communication between instances in your VPC when you use a different security group, you must add a rule like the following to your security groups.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
The ID of the security group	All	All	Allow inbound traffic from other instances assigned to this security group

Implementing Scenario 2

Use the following process to implement scenario 2 using the VPC wizard.

To implement scenario 2 using the VPC wizard

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **VPC Dashboard**.
3. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
4. Select the second option, **VPC with Public and Private Subnets**, and then click **Select**.
5. Verify the information on the confirmation page. Make any changes that you need, and then click **Create VPC** to create your VPC, subnets, Internet gateway, and route tables, and launch a NAT instance into the public subnet.

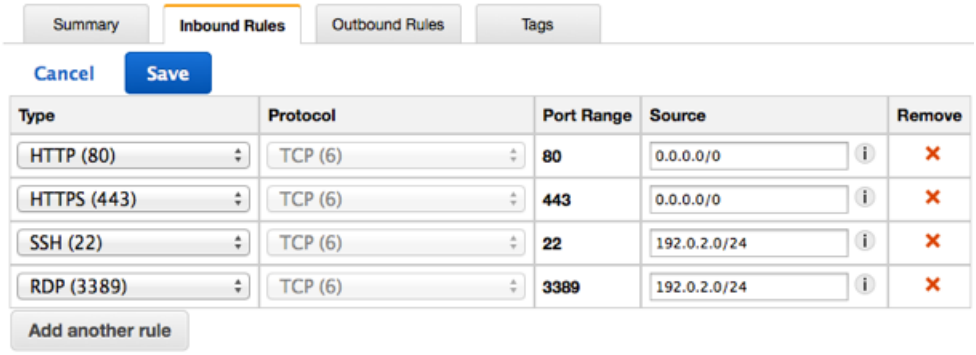
Because the WebServerSG and DBServerSG security groups reference each other, create all the security groups required for this scenario before you add rules to them.

To create the WebServerSG, NATSG, and DBServerSG security groups

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Security Groups**.
3. Click the **Create Security Group** button.
4. In the **Create Security Group** dialog box, specify `WebServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
5. Click the **Create Security Group** button again.
6. In the **Create Security Group** dialog box, specify `NATSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
7. Click the **Create Security Group** button again.
8. In the **Create Security Group** dialog box, specify `DBServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.

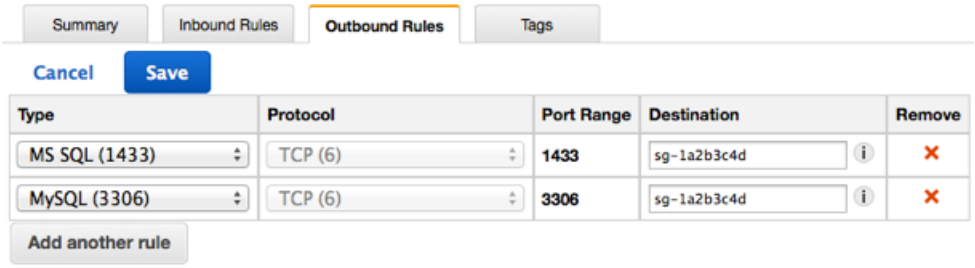
To add rules to the WebServerSG security group

1. Select the WebServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:
   a. Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
   b. Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
   c. Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   d. Click **Add another rule**, then select **RDP** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   e. Click **Save**.



3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:

   a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
   b. Select **MS SQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
   c. Click **Add another rule**, then select **MySQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
   d. Click **Save**.



To add the recommended rules to the NATSG security group

1. Select the NATSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:

   a. Select **HTTP** from the **Type** list, and enter the IP address range of your private subnet in the **Source** field.
   b. Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter the IP address range of your private subnet in the **Source** field.
   c. Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   d. Click **Save**.

3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:

   a. Locate the default rule that enables all outbound traffic, and then click **Remove**.

b. Select **HTTP** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
c. Click **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
d. Click **Save**.

To add the recommended rules to the DBServerSG security group

1. Select the DBServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:
   a. Select **MS SQL** from the **Type** list, and specify the ID of your WebServerSG security group in the **Source** field.
   b. Click **Add another rule**, then select **MYSQL** from the **Type** list, and specify the ID of your WebServerSG security group in the **Source** field.
   c. Click **Save**.
3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:
   a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
   b. Select **HTTP** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
   c. Click **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
   d. Click **Save**.

When the VPC wizard launched the NAT instance, it used the default security group for the VPC. You need to associate the NAT instance with the NATSG security group instead.

To change the security group of the NAT instance

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, click **Network Interfaces**.
3. Select the network interface for the NAT instance from the list, and then select **Change Security Groups** from the **Actions** list.
4. In the **Change Security Groups** dialog box, select the NATSG security group that you created (see Security for Scenario 2 (p. 25)) from the **Security groups** list, and then click **Save**.

You can launch instances into your VPC. If you're already familiar with launching instances outside a VPC, then you already know most of what you need to know to launch an instance into a VPC.

To launch an instance (web server or database server)

1. Create the WebServerSG and DBServerSG security groups if you haven't done so already (see Security for Scenario 2 (p. 25)). You'll specify one of these security groups when you launch the instance.
2. Start the launch wizard:
   a. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
   b. Click the **Launch Instance** button from the dashboard.

3. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
4. On the **Configure Instance Details** page, select the VPC that you created earlier from the **Network** list, and then select a subnet. For example, launch a web server into the public subnet and the database server into the private subnet.
5. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IP address. To be able to connect to your instance, you can assign a public IP address now, or allocate an Elastic IP address and assign it to your instance after it's launched. To assign a public IP address now, ensure that you select **Enable** from the **Auto-assign Public IP** list.

   Note
   You can only assign a public IP address to a single, new network interface with the device index of eth0. For more information, see Assigning a Public IP Address During Launch (p. 118).
6. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, select the **Select an existing security group** option, and select a security group for the instance (**WebServerSG** for a web server or **DBServerSG** for a database server). Click **Review and Launch**.
7. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.

If you did not assign a public IP address to your instance in step 5, you will not be able to connect to it. Before you can access an instance in your public subnet, you must assign it an Elastic IP address.

To allocate an Elastic IP address and assign it to an instance using the console

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. Click **Yes, Allocate**.

   Note
   If your account supports EC2-Classic, first choose **EC2-VPC** from the **Network platform** list.
5. Select the Elastic IP address from the list, and then click the **Associate Address** button.
6. In the **Associate Address** dialog box, select the network interface or instance. Select the address to associate the Elastic IP address with from the corresponding **Private IP address** list, and then click **Yes, Associate**.

You can now connect to your instances in the VPC. For information about how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access

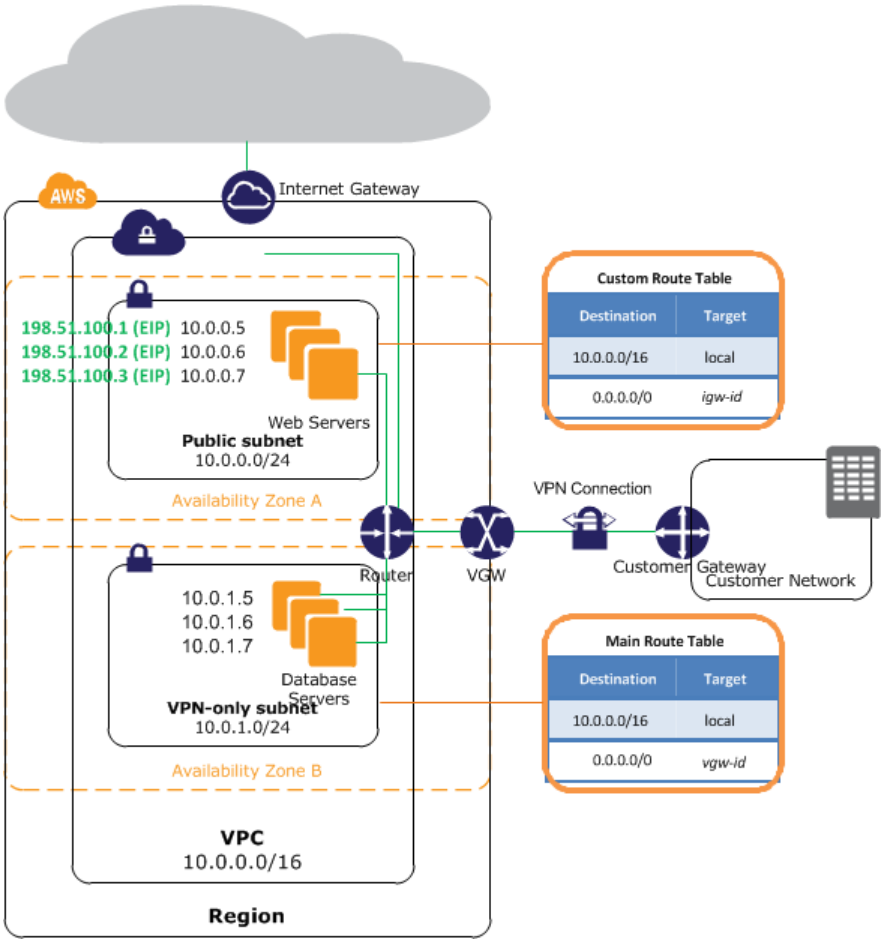
The configuration for this scenario includes a virtual private cloud (VPC) with a public subnet and a private subnet, and a virtual private gateway to enable communication with your own network over an IPsec VPN tunnel. We recommend this scenario if you want to extend your network into the cloud and also directly access the Internet from your VPC. This scenario enables you to run a multi-tiered application with a scalable web front end in a public subnet, and to house your data in a private subnet that is connected to your network by an IPsec VPN connection.

Topics

- Configuration for Scenario 3 (p. 32)
- Basic Configuration for Scenario 3 (p. 33)
- Routing for Scenario 3 (p. 33)
- Security for Scenario 3 (p. 35)
- Implementing Scenario 3 (p. 37)

Configuration for Scenario 3

The following diagram shows the key components of the configuration for this scenario.



Important
For this scenario, the *Amazon VPC Network Administrator Guide* describes what your network administrator needs to do to configure the Amazon VPC customer gateway on your side of the VPN connection.

Basic Configuration for Scenario 3

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A public subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- A VPN-only subnet of size /24 (example CIDR: 10.0.1.0/24). This provides 256 private IP addresses.
- An Internet gateway. This connects the VPC to the Internet and to other AWS products.
- A VPN connection between your VPC and your network. The VPN connection consists of a virtual private gateway located on the Amazon side of the VPN connection and a customer gateway located on your side of the VPN connection.
- Instances with private IP addresses in the subnet range (examples: 10.0.0.5 and 10.0.1.5), which enables the instances to communicate with each other and other instances in the VPC. Instances in the public subnet also have Elastic IP addresses (example: 198.51.100.1), which enables them to be reached from the Internet. Instances in the VPN-only subnet are back-end servers that don't need to accept incoming traffic from the Internet, but can send and receive traffic from your network.
- A custom route table associated with the public subnet. This route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC, and an entry that enables instances in the subnet to communicate directly with the Internet.
- The main route table associated with the VPN-only subnet. The route table contains an entry that enables instances in the subnet to communicate with other instances in the VPC, and an entry that enables instances in the subnet to communicate directly with your network.

For more information about subnets, see Your VPC and Subnets (p. 47) and IP Addressing in Your VPC (p. 116). For more information about Internet gateways, see Internet Gateways (p. 133). For more information about your VPN connection, see Adding a Hardware Virtual Private Gateway to Your VPC (p. 172). For more information about configuring a customer gateway, see the *Amazon VPC Network Administrator Guide*.

Routing for Scenario 3

Your VPC has an implied router (shown in the configuration diagram for this scenario). For this scenario, the VPC wizard updates the main route table used with the VPN-only subnet, and creates a custom route table and associates it with the public subnet. Otherwise, you'd need to create and associate the route tables yourself.

The instances in the VPN-only subnet can't reach the Internet directly; any Internet-bound traffic must first traverse the virtual private gateway to your network, where the traffic is then subject to your firewall and corporate security policies. If the instances send any AWS-bound traffic (for example, requests to the Amazon S3 or Amazon EC2 APIs), the requests must go over the virtual private gateway to your network and then egress to the Internet before reaching AWS.

Tip
Any traffic from your network going to an Elastic IP address for an instance in the public subnet goes over the Internet, and not over the virtual private gateway. You could instead set up a route and security group rules that enable the traffic to come from your network over the virtual private gateway to the public subnet.

The VPN connection is configured either as a statically-routed VPN connection or as a dynamically-routed VPN connection (using BGP). If you select static routing, you'll be prompted to manually enter the IP prefix for your network when you create the VPN connection. If you select dynamic routing, the IP prefix is advertised automatically to the virtual private gateway for your VPC using BGP.

The following tables describe the route tables for this scenario.

Main Route Table

The first row describes the entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic from the private subnet to your network over the virtual private gateway, which is specified using its AWS-assigned identifier (for example, `vgw-1a2b3c4d`).

Destination	Target
`10.0.0.0/16`	local
`0.0.0.0/0`	`vgw-`*`xxxxxxxx`*

Custom Route Table

The first row describes the entry for local routing in the VPC; this entry enables the instances in the VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic from the public subnet to the Internet over the Internet gateway, which is specified using its AWS-assigned identifier (for example, `igw-1a2b3c4d`).

Destination	Target
`10.0.0.0/16`	local
`0.0.0.0/0`	`igw-`*`xxxxxxxx`*

Alternate Routing

Alternatively, if you want instances in the private subnet to access the Internet, you could set up the routing so that the Internet-bound traffic for the subnet goes to a network address translation (NAT) instance in the public subnet. The NAT instance enables the instances in the VPN-only subnet to send requests over the Internet gateway (for example, for software updates). To enable the private subnet's Internet-bound traffic to go to the NAT instance, you must update the main route table as follows.

Main Route Table

The first row describes the entry for local routing in the VPC. The second row describes the entry for routing the subnet traffic bound for your network to the virtual private gateway, which is specified using its AWS-assigned identifier (for example, `vgw-1a2b3c4d`). The third row sends all other subnet traffic to the NAT instance, which is specified by its AWS-assigned identifier (for example, `i-1a2b3c4d`).

Destination	Target
`10.0.0.0/16`	local
`172.16.0.0/12`	`vgw-`*`xxxxxxxx`*
`0.0.0.0/0`	`i-`*`xxxxxxxx`*

For information about setting up a NAT instance manually, see NAT Instances (p. 138). For information about using the VPC wizard to set up a NAT instance, see Scenario 2: VPC with Public and Private Subnets (NAT) (p. 22).

Security for Scenario 3

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see Security in Your VPC (p. 62).

For scenario 3, you'll use security groups but not network ACLs. If you'd like to use a network ACL, see Recommended Rules for Scenario 3 (p. 83).

Topics

- Recommended Security Groups (p. 35)

Recommended Security Groups

Your VPC comes with a default security group whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances assigned to the security group. If you don't specify a security group when you launch an instance, the instance is automatically assigned to this default security group.

For this scenario, we recommend that you create the following security groups instead of modifying the default security group:

- **WebServerSG—**For the web servers in the public subnet
- **DBServerSG—**For the database servers in the VPN-only subnet

The instances assigned to a security group can be in different subnets. However, in this scenario, each security group corresponds to the type of role an instance plays, and each role requires the instance to be in a particular subnet. Therefore, in this scenario, all instances assigned to a security group are in the same subnet.

The WebServerSG security group is the security group that you'll specify when you launch your web servers into your public subnet. The following table describes the recommended rules for this security group, which allow the web servers to receive Internet traffic, as well as SSH and RDP traffic from your network. The web servers can also initiate read and write requests to the database server instances in the VPN-only subnet.

> **Note**
> The group includes both SSH and RDP access, and both Microsoft SQL Server and MySQL access. For your situation, you might only need rules for Linux (SSH and MySQL) or Windows (RDP and Microsoft SQL Server).

WebServerSG: Recommended Rules

Inbound			
Source	**Protocol**	**Port Range**	**Comments**

0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
Your network's public IP address range	TCP	22	Allow inbound SSH access to Linux instances from your network (over the Internet gateway)
Your network's public IP address range	TCP	3389	Allow inbound RDP access to Windows instances from your network (over the Internet gateway)
Outbound			
The ID of your DBServerSG security group	TCP	1433	Allow outbound Microsoft SQL Server access to the database servers assigned to DBServerSG
The ID of your DBServerSG security group	TCP	3306	Allow outbound MySQL access to the database servers assigned to DBServerSG

The DBServerSG security group is the security group that you'll specify when you launch your database servers into your VPN-only subnet. The following table describes the recommended rules for this security group, which allow Microsoft SQL Server and MySQL read and write requests from the web servers and SSH and RDP traffic from your network. The database servers can also initiate traffic bound for the Internet (your route table sends that traffic over the virtual private gateway).

DBServerSG: Recommended Rules

Inbound			
Source	**Protocol**	**Port range**	**Comments**
The ID of your WebServerSG security group	TCP	1433	Allow web servers assigned to WebServerSG Microsoft SQL Server access to database servers assigned to DBServerSG
The ID of your WebServerSG security group	TCP	3306	Allow web servers assigned to WebServerSG MySQL access to database servers assigned to DBServerSG
Your network's IP address range	TCP	22	Allow inbound SSH traffic to Linux instances from your network (over the virtual private gateway)
Your network's IP address range	TCP	3389	Allow inbound RDP traffic to Windows instances from your network (over the virtual private gateway)
Outbound			
Destination	**Protocol**	**Port range**	**Comments**

0.0.0.0/0	TCP	80	Allow outbound HTTP access to the Internet (for example, for software updates) over the virtual private gateway
0.0.0.0/0	TCP	443	Allow outbound HTTPS access to the Internet (for example, for software updates) over the virtual private gateway

The default security group for a VPC has rules that automatically allow assigned instances to communicate with each other. To allow that type of communication between instances in your VPC when you use a different security group, you must add a rule like the following to your security groups.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
The ID of the security group	All	All	Allow inbound traffic from other instances assigned to this security group

Implementing Scenario 3

Use the following process to implement scenario 3 using the VPC wizard.

To prepare your customer gateway

1. Determine the appliance you'll use as your customer gateway. For more information about the devices that we've tested, see Amazon Virtual Private Cloud FAQs. For more information about the requirements for your customer gateway, see the Amazon VPC Network Administrator Guide.
2. Obtain the Internet-routable IP address for the customer gateway's external interface. The address must be static and may be behind a device performing network address translation (NAT); however, NAT traversal (NAT-T) is not supported.
3. Gather the list of internal IP ranges (in CIDR notation) that should be advertised across the VPN connection to the virtual private gateway (if you are using a statically routed VPN connection). For more information, see VPN Routing Options (p. 174).

To implement scenario 3 using the VPC wizard

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **VPC Dashboard**.
3. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
4. Select the third option, **VPC with Public and Private Subnets and Hardware VPN Access**, and then click **Select**.
5. On the first page of the wizard, confirm the details for your VPC, public and private subnets, and then click **Next**.
6. On the **Configure your VPN** page, do the following, and then click **Create VPC**:
   - In **Customer Gateway IP**, specify the public IP address of your VPN router.
   - Optionally specify a name for your customer gateway and VPN connection.
   - In **Routing Type**, select one of the routing options as follows:
     - If your VPN router supports Border Gateway Protocol (BGP), select **Dynamic (requires BGP)**.

- If your VPN router does not support BGP, click **Static**. In **IP Prefix**, add each IP prefix for your network.

  For more information about which option to choose, see Amazon Virtual Private Cloud FAQs. For more information about dynamic versus static routing, see VPN Routing Options (p. 174).

7. When the wizard is done, click **VPN Connections** in the navigation pane. Select the VPN connection that the wizard created, and click **Download Configuration**. In the dialog box, select the vendor for the customer gateway, the platform, and the software version, and then click **Yes, Download**.
8. Save the text file containing the VPN configuration and give it to the network administrator along with this guide: Amazon VPC Network Administrator Guide. The VPN won't work until the network administrator configures the customer gateway.

Because the WebServerSG and DBServerSG security groups reference each other, create all the security groups required for this scenario before you add rules to them.

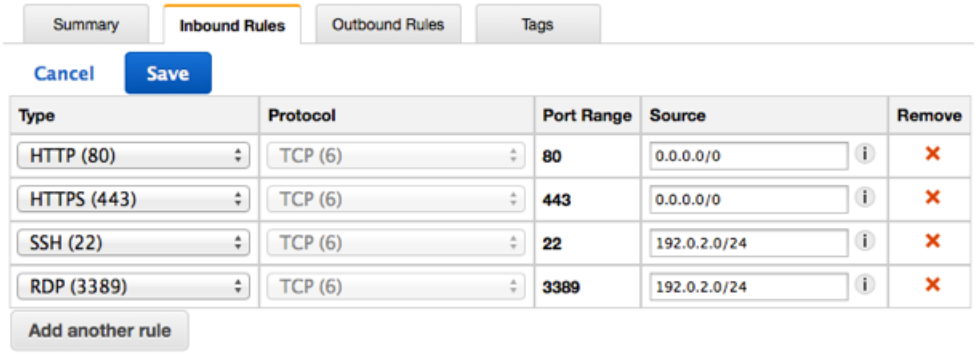
To create the WebServerSG and DBServerSG security groups

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Security Groups**.
3. Click the **Create Security Group** button.
4. In the **Create Security Group** dialog box, specify `WebServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
5. Click the **Create Security Group** button again.
6. In the **Create Security Group** dialog box, specify `DBServerSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.

To add the recommended rules to the WebServerSG security group

1. Select the WebServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:

   a. Select **HTTP** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
   b. Click **Add another rule**, then select **HTTPS** from the **Type** list, and enter `0.0.0.0/0` in the **Source** field.
   c. Click **Add another rule**, then select **SSH** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   d. Click **Add another rule**, then select **RDP** from the **Type** list. Enter your network's public IP address range in the **Source** field.
   e. Click **Save**.



3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:

   a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
   b. Select **MS SQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
   c. Click **Add another rule**, then select **MySQL** from the **Type** list. In the **Destination** field, specify the ID of the DBServerSG security group.
   d. Click **Save**.

To add the recommended rules to the DBServerSG security group

1. Select the DBServerSG security group that you created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
2. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:

   a. Select **SSH** from the **Type** list, and enter the IP address range of your network in the **Source** field.
   b. Click **Add another rule**, then select **RDP** from the **Type** list, and enter the IP address range of your network in the **Source** field.
   c. Click **Add another rule**, then select **MS SQL** from the **Type** list. Specify the ID of your WebServerSG security group in the **Source** field.
   d. Click **Add another rule**, then select **MYSQL** from the **Type** list. Specify the ID of your WebServerSG security group in the **Source** field.
   e. Click **Save**.

3. On the **Outbound Rules** tab, click **Edit** and add rules for outbound traffic as follows:

   a. Locate the default rule that enables all outbound traffic, and then click **Remove**.
   b. Select **HTTP** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
   c. Click **Add another rule**, then select **HTTPS** from the **Type** list. In the **Destination** field, enter `0.0.0.0/0`.
   d. Click **Save**.

After your network administrator configures your customer gateway, you can launch instances into your VPC. If you're already familiar with launching instances outside a VPC, then you already know most of what you need to know to launch an instance into a VPC.

To launch an instance (web server or database server)

1. Create the WebServerSG and DBServerSG security groups if you haven't done so already (see Security for Scenario 3 (p. 35)). You'll specify one of these security groups when you launch the instance.
2. Start the launch wizard:
   a. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
   b. Click the **Launch Instance** button from the dashboard.
3. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
4. On the **Configure Instance Details** page, select the VPC that you created earlier from the **Network** list, and then select a subnet. For example, launch a web server into the public subnet and the database server into the private subnet.
5. (Optional) By default, instances launched into a nondefault VPC are not assigned a public IP address. To be able to connect to your instance, you can assign a public IP address now, or allocate an Elastic IP address and assign it to your instance after it's launched. To assign a public IP address now, ensure that you select **Enable** from the **Auto-assign Public IP** list.

   Note
   You can only assign a public IP address to a single, new network interface with the device index of eth0. For more information, see Assigning a Public IP Address During Launch (p. 118).
6. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, select the **Select an existing security group** option, and select a security group for the instance (**WebServerSG** for a web server or **DBServerSG** for a database server). Click **Review and Launch**.
7. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.

For the instances running in the VPN-only subnet, you can test their connectivity by pinging them from your network. For more information, see Testing the End-to-End Connectivity of Your Instance (p. 180).

If you did not assign a public IP address to your instance in step 5, you will not be able to connect to it. Before you can access an instance in your public subnet, you must assign it an Elastic IP address.

To allocate an Elastic IP address and assign it to an instance using the console

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. Click **Yes, Allocate**.

   Note
   If your account supports EC2-Classic, first choose **EC2-VPC** from the **Network platform** list.
5. Select the Elastic IP address from the list, and then click the **Associate Address** button.
6. In the **Associate Address** dialog box, select the network interface or instance. Select the address to associate the Elastic IP address with from the corresponding **Private IP address** list, and then click **Yes, Associate**.

In scenario 3, you need a DNS server that enables your public subnet to communicate with servers on the Internet, and you need another DNS server that enables your VPN-only subnet to communicate with servers in your network.

Your VPC automatically has a set of DHCP options with domain-name-servers=AmazonProvidedDNS. This is a DNS server that Amazon provides to enable any public subnets in your VPC to communicate with the Internet over an Internet gateway. You must provide your own DNS server and add it to the list of DNS servers your VPC uses. Sets of DHCP options aren't modifiable, so you must create a set of DHCP options that includes both your DNS server and the Amazon DNS server, and update the VPC to use the new set of DHCP options.

To update the DHCP options

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **DHCP Options Sets**.
3. Click the **Create DHCP Options Set** button.
4. In the **Create DHCP Options Set** dialog box, in the **Domain name servers** box, specify the address of the Amazon DNS server (AmazonProvidedDNS) and the address of your DNS server, separated by a comma, and then click **Yes, Create**. In this example, your DNS server is 192.0.2.1.
5. In the navigation pane, click **Your VPCs**.
6. Select the VPC, and then click the **Edit** button in the **Summary** tab.
7. Select the ID of the new set of options from the **DHCP options set** list and then click **Save**.
8. (Optional) The VPC now uses this new set of DHCP options and therefore has access to both DNS servers. If you want, you can delete the original set of options that the VPC used.

You can now connect to your instances in the VPC. For information about how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access

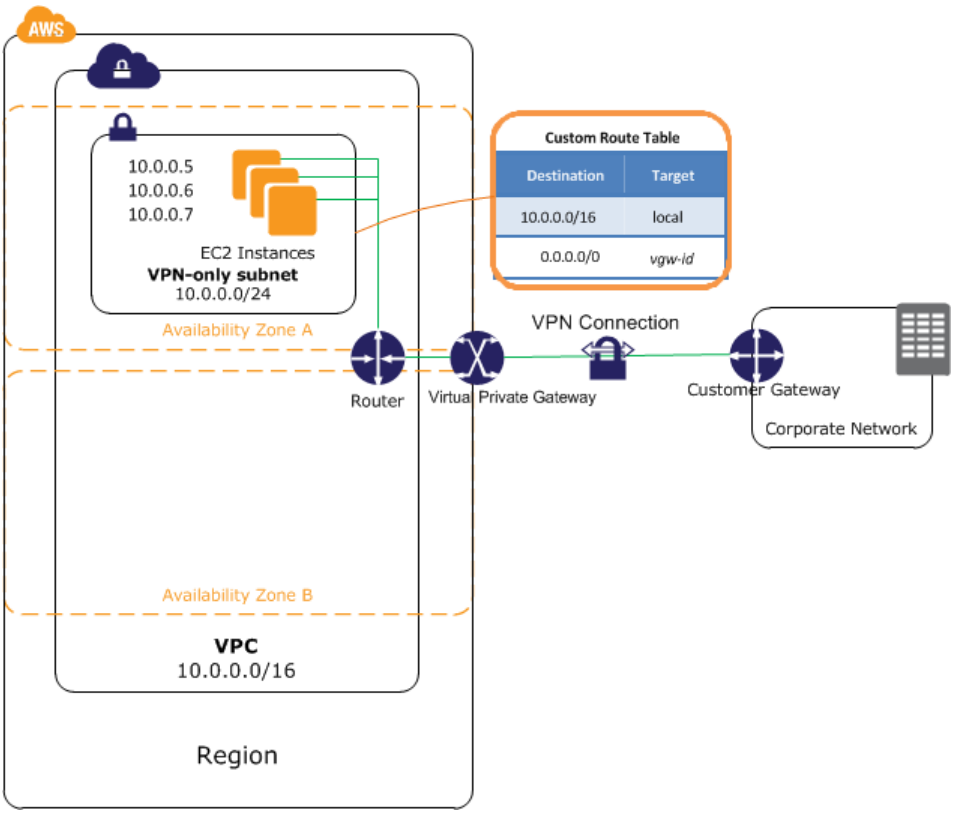
The configuration for this scenario includes a virtual private cloud (VPC) with a single private subnet, and a virtual private gateway to enable communication with your own network over an IPsec VPN tunnel. There is no Internet gateway to enable communication over the Internet. We recommend this scenario if you want to extend your network into the cloud using Amazon's infrastructure without exposing your network to the Internet.

Topics

- Configuration for Scenario 4 (p. 41)
- Basic Components for Scenario 4 (p. 42)
- Routing for Scenario 4 (p. 43)
- Security for Scenario 4 (p. 43)
- Implementing Scenario 4 (p. 44)

Configuration for Scenario 4

The following diagram shows the key components of the configuration for this scenario.



Important
For this scenario, the *Amazon VPC Network Administrator Guide* describes what your network administrator needs to do to configure the Amazon VPC customer gateway on your side of the VPN connection.

Basic Components for Scenario 4

The following list describes the basic components presented in the configuration diagram for this scenario:

- A virtual private cloud (VPC) of size /16 (example CIDR: 10.0.0.0/16). This provides 65,536 private IP addresses.
- A VPN-only subnet of size /24 (example CIDR: 10.0.0.0/24). This provides 256 private IP addresses.
- A VPN connection between your VPC and your network. The VPN connection consists of a virtual private gateway located on the Amazon side of the VPN connection and a customer gateway located on your side of the VPN connection.
- Instances with private IP addresses in the subnet range (examples: 10.0.0.5, 10.0.0.6, and 10.0.0.7), which enables the instances to communicate with each other and other instances in the VPC.
- A route table entry that enables instances in the subnet to communicate with other instances in the VPC, and a route table entry that enables instances in the subnet to communicate directly with your network.

For more information about subnets, see Your VPC and Subnets (p. 47) and IP Addressing in Your VPC (p. 116). For more information about your VPN connection, see Adding a Hardware Virtual Private Gateway to Your VPC (p. 172). For more information about configuring a customer gateway, see the *Amazon VPC Network Administrator Guide*.

Routing for Scenario 4

Your VPC has an implied router (shown in the configuration diagram for this scenario.) For this scenario, the VPC wizard creates a route table that routes all traffic destined for an address outside the VPC to the VPN connection, and associates the route table with the subnet. Otherwise, you'd need to create and associate the route table yourself.

The following table shows what the route table looks like for the example addresses used in the configuration diagram for this scenario. The first row describes the entry for local routing in the VPC; this entry enables the instances in this VPC to communicate with each other. The second row describes the entry for routing all other subnet traffic to the virtual private gateway, which is specified using its AWS-assigned identifier (for example, `vgw-1a2b3c4d`).

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	vgw-*xxxxxxxx*

The VPN connection is configured either as a statically-routed VPN connection or as a dynamically routed VPN connection (using BGP). If you select static routing, you'll be prompted to manually enter the IP prefix for your network when you create the VPN connection. If you select dynamic routing, the IP prefix is advertised automatically to your VPC through BGP.

The instances in your VPC can't reach the Internet directly; any Internet-bound traffic must first traverse the virtual private gateway to your network, where the traffic is then subject to your firewall and corporate security policies. If the instances send any AWS-bound traffic (for example, requests to Amazon S3 or Amazon EC2), the requests must go over the virtual private gateway to your network and then to the Internet before reaching AWS.

Security for Scenario 4

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see Security in Your VPC (p. 62).

For scenario 4, you'll use the default security group for your VPC but not network ACLs. If you'd like to use a network ACL, see Recommended Rules for Scenario 4 (p. 86).

Recommended Security Group Rules

Your VPC comes with a default security group whose initial settings deny all inbound traffic, allow all outbound traffic, and allow all traffic between the instances assigned to the security group. We recommend that you add inbound rules to the default security group to allow SSH traffic (Linux) and Remote Desktop traffic (Windows) from your network.

Important
The default security group automatically allows assigned instances to communicate with each other, so you don't have to add a rule to allow this. If you use a different security group, you must add a rule to allow this.

The following table describes the inbound rules that you should add to the default security group for your VPC.

Default Security Group: Recommended Rules

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
Private IP address range of your network	TCP	22	(Linux instances) Allow inbound SSH traffic from your network
Private IP address range of your network	TCP	3389	(Windows instances) Allow inbound RDP traffic from your network

Implementing Scenario 4

Use the following process to implement scenario 4 using the VPC wizard.

To prepare your customer gateway

1. Determine the appliance you'll use as your customer gateway. For information about the devices that we've tested, see Amazon Virtual Private Cloud FAQs. For more information about the requirements for your customer gateway, see the *Amazon VPC Network Administrator Guide*.
2. Obtain the Internet-routable IP address for the customer gateway's external interface. The address must be static and may be behind a device performing network address translation (NAT); however, NAT traversal (NAT-T) is not supported.
3. Gather the list of internal IP ranges (in CIDR notation) that should be advertised across the VPN connection to the virtual private gateway (if you are using a statically routed VPN connection). For more information, see VPN Routing Options (p. 174).

Next, use the VPC wizard as described in the following procedure to create your VPC and a VPN connection.

To implement scenario 4 using the VPC wizard

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **VPC Dashboard**.
3. Locate the **Your Virtual Private Cloud** area of the dashboard and click **Get started creating a VPC**, if you have no VPC resources, or click **Start VPC Wizard**.
4. Select the fourth option, **VPC with a Private Subnet Only and Hardware VPN Access**, and then click **Select**.
5. On the first page of the wizard, confirm the details for your VPC and private subnet, and then click **Next**.
6. On the **Configure your VPN** page, do the following, and then click **Create VPC**:
   - In **Customer Gateway IP**, specify the public IP address of your VPN router.
   - Optionally specify a name for your customer gateway and VPN connection.
   - In **Routing Type**, select one of the routing options as follows:
     - If your VPN router supports Border Gateway Protocol (BGP), select **Dynamic (requires BGP)**.
     - If your VPN router does not support BGP, click **Static**. In **IP Prefix**, add each IP prefix for your network.

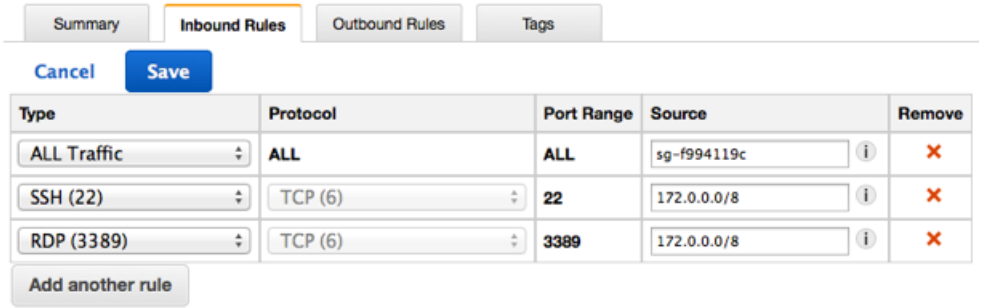
For more information about which option to choose, see Amazon Virtual Private Cloud FAQs. For more information about dynamic versus static routing, see VPN Routing Options (p. 174).

7. When the wizard is done, click **VPN Connections** in the navigation pane. Select the VPN connection that the wizard created, and click **Download Configuration**. In the dialog box, select the vendor for the customer gateway, the platform, and the software version, and then click **Yes, Download**.
8. Save the text file containing the VPN configuration and give it to the network administrator along with this guide: Amazon VPC Network Administrator Guide. The VPN won't work until the network administrator configures the customer gateway.

For this scenario, you need to update the default security group with new inbound rules that allow SSH and Remote Desktop (RDP) access from your network. If the instances won't initiate outbound communication, we can also remove the default outbound rule. Reminder: the initial settings of the default security group block all inbound traffic, allow all outbound traffic, and allow instances assigned to the group to communicate with each other.

To update the rules for the default security group

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. Click **Security Groups** in the navigation pane, and then select the default security group for the VPC. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
3. On the **Inbound Rules** tab, click **Edit** and add rules for inbound traffic as follows:
   a. Select **SSH** from the **Type** list, and enter your network's private IP address range in the **Source** field.
   b. Click **Add another rule**, then select **RDP** from the **Type** list, and enter your network's private IP address range in the **Source** field.
   c. Click **Save**.



4. On the **Outbound Rules** tab, click **Edit**, locate the default rule that enables all outbound traffic, click **Remove**, and then click **Save**.

After your network administrator configures your customer gateway, you can launch instances into your VPC. If you're already familiar with launching instances outside a VPC, then you already know most of what you need to know to launch an instance into a VPC.

To launch an instance

1. Start the launch wizard:

a. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
b. Click the **Launch Instance** button from the dashboard.

2. Follow the directions in the wizard. Choose an AMI, choose an instance type, and then click **Next: Configure Instance Details**.
3. On the **Configure Instance Details** page, select the VPC that you created earlier from the **Network** list, and then select a subnet. Click **Next: Add Storage**.
4. On the next two pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, select the **Select an existing security group** option, and select the default security group. Click **Review and Launch**.
5. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a keypair and launch your instance.

In scenario 4, you need a DNS server that enables your VPN-only subnet to communicate with servers in your network. You must create a new set of DHCP options that includes your DNS server and then configure the VPC to use that set of options.

Note
Your VPC automatically has a set of DHCP options with domain-name-servers=AmazonProvidedDNS. This is a DNS server that Amazon provides to enable any public subnets in your VPC to communicate with the Internet over an Internet gateway. Scenario 4 doesn't have any public subnets, so you don't need this set of DHCP options.

To update the DHCP options

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **DHCP Options Sets**.
3. Click the **Create DHCP Options Set** button.
4. In the **Create DHCP Options Set** dialog box, in the **Domain name servers** box, enter the address of your DNS server, and then click **Yes, Create**. In this example, your DNS server is 192.0.2.1.
5. In the navigation pane, click **Your VPCs**.
6. Select the VPC, and then click the **Edit** button in the **Summary** tab.
7. Select the ID of the new set of options from the **DHCP options set** list and then click **Save**.
8. (Optional) The VPC now uses this new set of DHCP options and therefore uses your DNS server. If you want, you can delete the original set of options that the VPC used.

You can now use SSH or RDP to connect to your instance in the VPC. For information about how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Your VPC and Subnets

To get started with Amazon Virtual Private Cloud (Amazon VPC), you'll create a VPC and subnets. For a general overview of VPCs and subnets, see What is Amazon VPC? (p. 1).

Topics

- Your VPC (p. 47)
- Subnets in Your VPC (p. 50)
- CLI Overview (p. 55)

Your VPC

A virtual private cloud (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. When you create a VPC, you specify the set of IP addresses for the VPC in the form of a Classless Inter-Domain Routing (CIDR) block (for example, `10.0.0.0/16`). For more information about CIDR notation and what "`/16`" means, see Classless Inter-Domain Routing on Wikipedia.

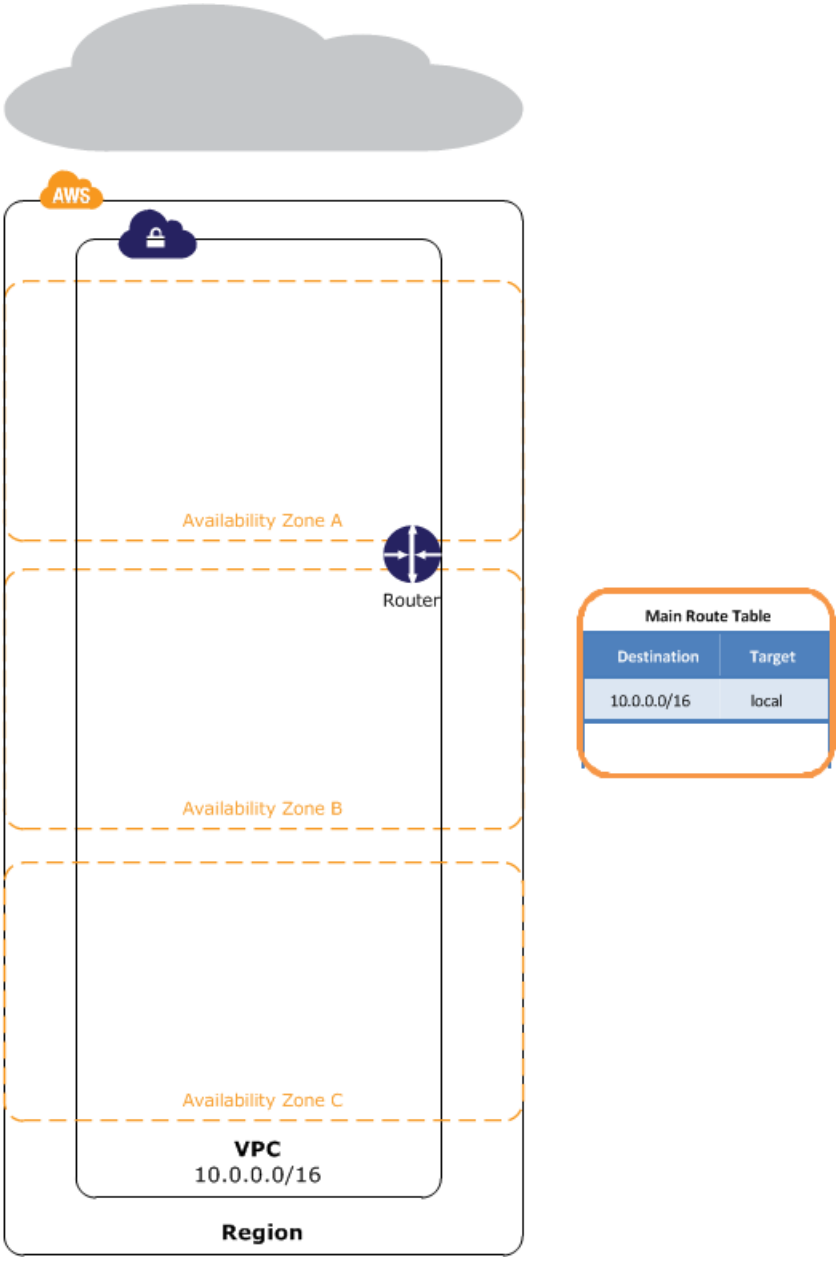
For information about the number of VPCs that you can create, see Amazon VPC Limits (p. 194).

Topics

- Your New VPC (p. 47)
- VPC Sizing (p. 48)
- Connections with Your Local Network and Other VPCs (p. 49)
- Creating a VPC (p. 49)
- Deleting Your VPC (p. 50)

Your New VPC

The following diagram shows a new VPC with a default route table.



You need to add a subnet before you can launch an instance into your VPC.

VPC Sizing

You can assign a single CIDR block to a VPC. The allowed block size is between a `/28` netmask and `/16` netmask. In other words, the VPC can contain from 16 to 65,536 IP addresses. You can't change the size of a VPC after you create it. If your VPC is too small to meet your needs, create a new, larger VPC, and then migrate your instances to the new VPC. To do this, create AMIs from your running instances, and then launch replacement instances in your new, larger VPC. You can then terminate your old instances, and delete your smaller VPC. For more information, see Deleting Your VPC (p. 50).

Connections with Your Local Network and Other VPCs

You can optionally set up a connection between your VPC and your corporate or home network. If you have an IP address prefix in your VPC that overlaps with one of your networks' prefixes, any traffic to the network's prefix is dropped. For example, let's say that you have the following:

- A VPC with CIDR block `10.0.0.0/16`
- A subnet in that VPC with CIDR block `10.0.1.0/24`
- Instances running in that subnet with IP addresses `10.0.1.4` and `10.0.1.5`
- On-premises host networks using CIDR blocks `10.0.37.0/24` and `10.1.38.0/24`

When those instances in the VPC try to talk to hosts in the `10.0.37.0/24` address space, the traffic is dropped because `10.0.37.0/24` is part of the larger prefix assigned to the VPC (`10.0.0.0/16`). The instances can talk to hosts in the `10.1.38.0/24` space because that block isn't part of `10.0.0.0/16`.

You can also create a VPC peering connection between your VPCs, or with a VPC in another AWS account. A VPC peering connection enables you to route traffic between the VPCs using private IP addresses; however, you cannot create a VPC peering connection between VPCs that have overlapping CIDR blocks. For more information, see VPC Peering (p. 152).

We therefore recommend that you create a VPC with a CIDR range large enough for expected future growth, but not one that overlaps with current or expected future subnets anywhere in your corporate or home network, or that overlaps with current or future VPCs.

Creating a VPC

There are two ways to create a VPC using the Amazon VPC console: the **Create VPC** dialog box and the VPC wizard. The following procedure uses the **Create VPC** dialog box, which creates only the VPC; you'd need to subsequently add subnets, gateways, and routing tables. For information about using the VPC wizard to create a VPC plus its subnets, gateways, and routing tables in one step, see VPC Wizard Scenarios for Amazon VPC (p. 17).

> **Note**
> (EC2-Classic) If you use the launch wizard in the Amazon EC2 console to launch a T2 instance type and you do not have any existing VPCs, the wizard creates a nondefault VPC for you, with a subnet in each Availability Zone, an Internet gateway, and a route table that routes all VPC traffic to the Internet gateway. For more information about T2 instance types, see T2 Instances in the *Amazon EC2 User Guide for Linux Instances*.

To create a VPC

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Your VPCs**.
3. Choose **Create VPC**.
4. In the **Create VPC** dialog box, specify the following VPC details as necessary, then choose **Yes, Create**.

   - Optionally provide a name for your VPC. Doing so creates a tag with a key of `Name` and the value that you specify.
   - Specify a CIDR block for the VPC. We recommend that you specify a CIDR block from the private (non-publicly routable) IP address ranges as specified in RFC 1918; for example, `10.0.0.0/16`, or `192.168.0.0/16`. It's possible to specify a range of publicly routable IP addresses; however, we currently do not support direct access to the Internet from publicly routable CIDR blocks in a

VPC. Windows instances cannot boot correctly if launched into a VPC with ranges from `224.0.0.0` to `255.255.255.255` (Class D and Class E IP address ranges). For more information about IP addresses, see IP Addressing in Your VPC (p. 116).

- Select a tenancy option, for example, a dedicated tenancy that ensures your instances run on single-tenant hardware. For more information about dedicated instances, see Dedicated Instances (p. 188).

Deleting Your VPC

You can delete your VPC at any time (for example, if you decide it's too small). However, you must terminate all instances in the VPC first. When you delete a VPC using the VPC console, we delete all its components, such as subnets, security groups, network ACLs, route tables, Internet gateways, VPC peering connections, and DHCP options.

If you have a VPN connection, you don't have to delete it or the other components related to the VPN (such as the customer gateway and virtual private gateway). If you plan to use the customer gateway with another VPC, we recommend you keep the VPN connection and the gateways. Otherwise, your network administrator must configure the customer gateway again after you create a new VPN connection.

To delete your VPC

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. Terminate all instances in the VPC. For more information, see Terminate Your Instance in the *EC2 User Guide*.
3. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
4. In the navigation pane, choose **Your VPCs**.
5. Select the VPC to delete, and then choose**Actions**, and select **Delete VPC**.
6. If you need to delete the VPN connection, select the option to do so; otherwise, leave it unselected. Choose **Yes, Delete**.

Subnets in Your VPC

You can create a VPC that spans multiple Availability Zones. For more information, see Creating a VPC (p. 49). After creating a VPC, you can add one or more subnets in each Availability Zone. Each subnet must reside entirely within one Availability Zone and cannot span zones. Availability Zones are distinct locations that are engineered to be isolated from failures in other Availability Zones. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location. AWS assigns a unique ID to each subnet.

For information about the number of subnets that you can create, see Amazon VPC Limits (p. 194).

Topics

- Your VPC with Subnets (p. 51)
- Subnet Sizing (p. 52)
- Subnet Routing (p. 53)
- Subnet Security (p. 53)
- Adding a Subnet to Your VPC (p. 54)
- Launching an Instance into Your Subnet (p. 54)
- Deleting Your Subnet (p. 54)

Your VPC with Subnets

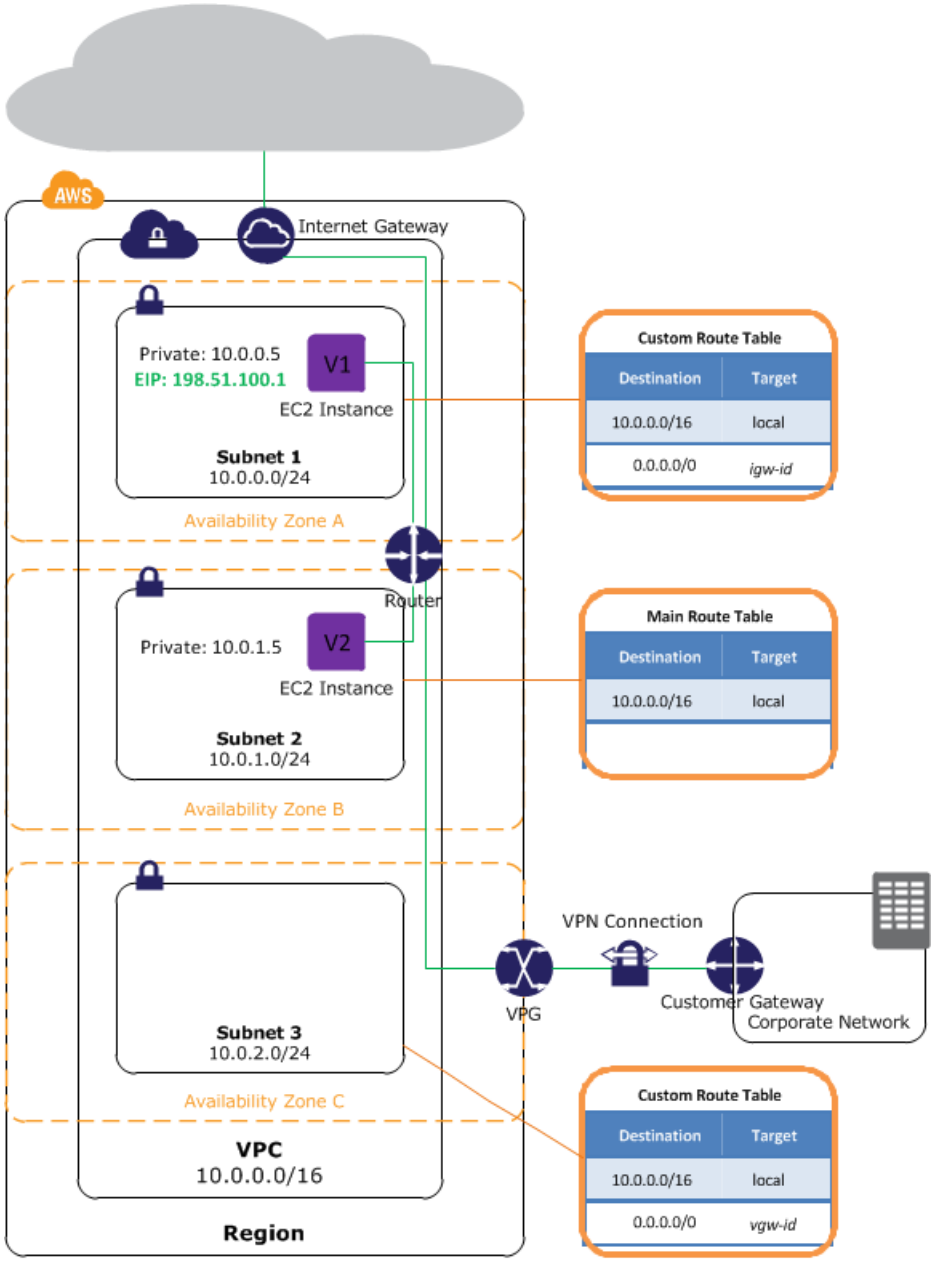
The following diagram shows a VPC that has been configured with subnets in multiple Availability Zones. You can optionally add an Internet gateway to enable communication over the Internet, or a virtual private network (VPN) connection to enable communication with your network, as shown in the diagram.

If a subnet's traffic is routed to an Internet gateway, the subnet is known as a *public subnet*. In this diagram, subnet 1 is a public subnet. If you want your instance in a public subnet to communicate with the Internet, it must have a public IP address or an Elastic IP address. For more information about public IP addresses, see Public and Private IP Addresses (p. 116).

If a subnet doesn't have a route to the Internet gateway, the subnet is known as a *private subnet*. In this diagram, subnet 2 is a private subnet.

If a subnet doesn't have a route to the Internet gateway, but has its traffic routed to a virtual private gateway, the subnet is known as a *VPN-only subnet*. In this diagram, subnet 3 is a VPN-only subnet.

> **Note**
> Regardless of the type of subnet, the internal IP address range of the subnet is always private — we do not announce the address block to the Internet. For more information, see IP Addressing in Your VPC (p. 116).



For more information, see VPC Wizard Scenarios for Amazon VPC (p. 17), Internet Gateways (p. 133), or Adding a Hardware Virtual Private Gateway to Your VPC (p. 172).

Subnet Sizing

When you create a subnet, you specify the CIDR block for the subnet. The CIDR block of a subnet can be the same as the CIDR block for the VPC (for a single subnet in the VPC), or a subset (to enable multiple subnets). The allowed block size is between a `/28` netmask and `/16` netmask. If you create more than one subnet in a VPC, the CIDR blocks of the subnets must not overlap.

For example, if you create a VPC with CIDR block `10.0.0.0/24`, it supports 256 IP addresses. You can break this CIDR block into two subnets, each supporting 128 IP addresses. One subnet uses CIDR block

10.0.0.0/25 (for addresses 10.0.0.0 - 10.0.0.127) and the other uses CIDR block 10.0.0.128/25 (for addresses 10.0.0.128 - 10.0.0.255).

There are many tools available to help you calculate subnet CIDR blocks. For information about a commonly used tool, see http://www.subnet-calculator.com/cidr.php. Also, your network engineering group can help you determine the CIDR blocks to specify for your subnets.

Important
AWS reserves both the first four IP addresses and the last IP address in each subnet CIDR block; they're not available for you to use. For example, in a subnet with CIDR block 10.0.0.0/24, the following IP addresses are reserved: 10.0.0.0, 10.0.0.1, 10.0.0.2, 10.0.0.3, and 10.0.0.255.

Subnet Routing

By design, each subnet must be associated with a route table, which specifies the allowed routes for outbound traffic leaving the subnet. Every subnet that you create is automatically associated with the main route table for the VPC. You can change the association, and you can change the contents of the main route table. For more information, see Route Tables (p. 123).

In the previous diagram, the route table associated with subnet 1 routes all traffic (0.0.0.0/0) to an Internet gateway (for example, igw-1a2b3c4d). Because instance V1 has an Elastic IP address, it can be reached from the Internet.

Note
The Elastic IP address or public IP address that's associated with your instance is accessed through the Internet gateway of your VPC. Traffic that goes through a VPN connection between your instance and another network traverses a virtual private gateway, not the Internet gateway, and therefore does not access the Elastic IP address or public IP address.

The instance V2 can't reach the Internet, but can reach other instances in the VPC. You can allow an instance in your VPC to initiate outbound connections to the Internet but prevent unsolicited inbound connections from the Internet using a network address translation (NAT) instance. Because you can allocate a limited number of Elastic IP addresses, we recommend that you use a NAT instance if you have more instances that require a static public IP address. For more information, see NAT Instances (p. 138).

The route table associated with subnet 3 routes all traffic (0.0.0.0/0) to a virtual private gateway (for example, vgw-1a2b3c4d).

Subnet Security

AWS provides two features that you can use to increase security in your VPC: *security groups* and *network ACLs*. Both features enable you to control the inbound and outbound traffic for your instances, but security groups work at the instance level, while network ACLs work at the subnet level. Security groups alone can meet the needs of many VPC users. However, some VPC users decide to use both security groups and network ACLs to take advantage of the additional layer of security that network ACLs provide. For more information about security groups and network ACLs and how they differ, see Security in Your VPC (p. 62).

By design, each subnet must be associated with a network ACL. Every subnet that you create is automatically associated with the VPC's default network ACL. You can change the association, and you can change the contents of the default network ACL. For more information, see Network ACLs (p. 70).

You can create a flow log on your VPC or subnet to capture the traffic that flows to and from the network interfaces in your VPC or subnet. You can also create a flow log on an individual network interface. Flow logs are published to CloudWatch Logs. For more information, see VPC Flow Logs (p. 106).

Adding a Subnet to Your VPC

When you add a new subnet to your VPC, you must set up the routing and security that you want for the subnet. You can do this manually, as described in this section, or let the VPC wizard set things up for you, as described in VPC Wizard Scenarios for Amazon VPC (p. 17).

To add a subnet to your VPC

1. Create the subnet.
   a. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
   b. In the navigation pane, choose **Subnets**.
   c. Choose **Create Subnet**.
   d. In the **Create Subnet** dialog box, optionally name your subnet, and then select the VPC, select the Availability Zone, specify the CIDR range for the subnet, and then choose **Yes, Create**.
2. Set up routing for the subnet. For example, you can add a route to an Internet gateway or a NAT instance. For more information, see Route Tables (p. 123).
3. (Optional) Create or modify your security groups as needed. For more information, see Security Groups for Your VPC (p. 64).
4. (Optional) Create or modify your network ACLs as needed. For more information about network ACLs, see Network ACLs (p. 70).

Launching an Instance into Your Subnet

To launch an instance into your subnet

1. Start the launch wizard:
   a. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
   b. On the dashboard, choose **Launch Instance**.
2. Follow the directions in the wizard. Select an AMI, choose an instance type, and then choose **Next: Configure Instance Details**.
3. On the **Configure Instance Details** page, ensure you have selected the required VPC in the **Network** list, then select the subnet to launch the instance into. Keep the other default settings on this page and choose **Next: Add Storage**.
4. On the next pages of the wizard, you can configure storage for your instance, and add tags. On the **Configure Security Group** page, choose from any existing security group that you own, or follow the wizard directions to create a new security group. Choose **Review and Launch** when you're done.
5. Review your settings and choose **Launch**.
6. Choose an existing key pair that you own, or create a new one, then choose **Launch Instances** when you're done.

Deleting Your Subnet

You must terminate any instances in the subnet first.

To delete your subnet

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. Terminate all instances in the subnet. For more information, see Terminate Your Instance in the *EC2 User Guide*.
3. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
4. In the navigation pane, choose **Subnets**.
5. Select the subnet to delete, choose **Subnet Actions**, and then choose **Delete**.
6. In the **Delete Subnet** dialog box, choose **Yes, Delete**.

CLI Overview

You can perform the tasks described on this page using a command line interface (CLI). For more information, including a list of available API actions, see Accessing Amazon VPC (p. 6).

Create a VPC

- create-vpc (AWS CLI)
- ec2-create-vpc (Amazon EC2 CLI)
- New-EC2Vpc (AWS Tools for Windows PowerShell)

Create a Subnet

- create-subnet (AWS CLI)
- ec2-create-subnet (Amazon EC2 CLI)
- New-EC2Subnet (AWS Tools for Windows PowerShell)

Describe a VPC

- describe-vpcs (AWS CLI)
- ec2-describe-vpcs (Amazon EC2 CLI)
- Get-EC2Vpc (AWS Tools for Windows PowerShell)

Describe a Subnet

- describe-subnets (AWS CLI)
- ec2-describe-subnets (Amazon EC2 CLI)
- Get-EC2Subnet (AWS Tools for Windows PowerShell)

Delete a VPC

- delete-vpc (AWS CLI)
- ec2-delete-vpc (Amazon EC2 CLI)
- Remove-EC2Vpc (AWS Tools for Windows PowerShell)

Delete a Subnet

- delete-subnet (AWS CLI)
- ec2-delete-subnet (Amazon EC2 CLI)

- Remove-EC2Subnet (AWS Tools for Windows PowerShell)

Your Default VPC and Subnets

If you created your AWS account after 2013-12-04, it supports only EC2-VPC. In this case, you'll have a *default VPC* in each AWS region. A default VPC is ready for you to use — you can immediately start launching instances into your default VPC without having to perform any additional configuration steps. A default VPC combines the benefits of the advanced networking features provided by the EC2-VPC platform with the ease of use of the EC2-Classic platform.

For more information about the EC2-Classic and EC2-VPC platforms, see Supported Platforms.

Topics

- Default VPC Basics (p. 57)
- Detecting Your Supported Platforms and Whether You Have a Default VPC (p. 59)
- Launching an EC2 Instance into Your Default VPC (p. 60)
- Deleting Your Default VPC (p. 61)

Default VPC Basics

This section provides information about your default virtual private cloud (VPC) and its default subnets.

Availability

If you created your AWS account after 2013-12-04, it supports only EC2-VPC. In this case, we create a default VPC for you in each AWS region. Therefore, unless you create a nondefault VPC and specify it when you launch an instance, we launch your instances into your default VPC.

If you created your AWS account before 2013-03-18, it supports both EC2-Classic and EC2-VPC in regions that you've used before, and only EC2-VPC in regions that you haven't used. In this case, we create a default VPC in each region in which you haven't created any AWS resources. Therefore, unless you create a nondefault VPC and specify it when you launch an instance in a region that you haven't used before, we launch the instance into your default VPC for that region. However, if you launch an instance in a region that you've used before, we launch the instance into EC2-Classic.

If you created your AWS account between 2013-03-18 and 2013-12-04, it may support only EC2-VPC, or it may support both EC2-Classic and EC2-VPC in some of the regions that you've used. For information about detecting the platform support in each region for your AWS account, see Detecting Your Supported Platforms and Whether You Have a Default VPC (p. 59). For information about when each region was

enabled for default VPCs, see Announcement: Enabling regions for the default VPC feature set in the AWS forum for Amazon VPC.

If an AWS account supports only EC2-VPC, any IAM accounts associated with this AWS account also support only EC2-VPC, and use the same default VPC as the AWS account.

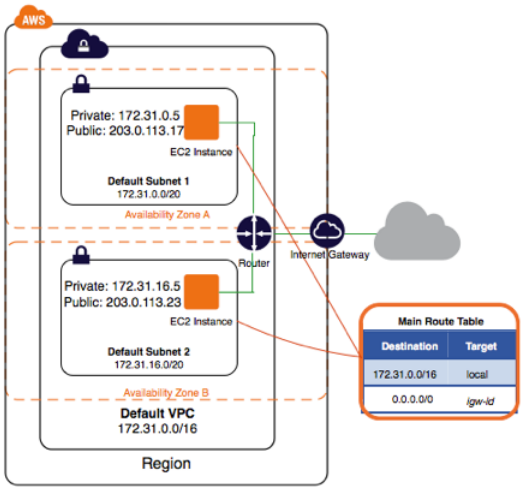
If your AWS account supports both EC2-Classic and EC2-VPC and you want the benefits of using EC2-VPC with the simplicity of launching instances into EC2-Classic, you can either create a new AWS account or launch your instances into a region that you haven't used before. If you'd prefer to add a default VPC to a region that doesn't have one, see "I really want a default VPC for my existing EC2 account. Is that possible?" in the Default VPCs FAQ.

Components

When we create a default VPC, we do the following to set it up for you:

- Create a default subnet in each Availability Zone.
- Create an Internet gateway and connect it to your default VPC.
- Create a main route table for your default VPC with a rule that sends all traffic destined for the Internet to the Internet gateway.
- Create a default security group and associate it with your default VPC.
- Create a default network access control list (ACL) and associate it with your default VPC.
- Associate the default DHCP options set for your AWS account with your default VPC.

The following figure illustrates the key components that we set up for a default VPC.



Instances that you launch into a default subnet receive both a public IP address and a private IP address. Instances in a default subnet also receive both public and private DNS hostnames. Instances that you launch into a nondefault subnet in a default VPC don't receive a public IP address or a DNS hostname. You can change your subnet's default public IP addressing behavior. For more information, see Modifying Your Subnet's Public IP Addressing Behavior (p. 117).

You can use a default VPC as you would use any other VPC; you can add subnets, modify the main route table, add additional route tables, associate additional security groups, update the rules of the default security group, and add VPN connections. You can also create additional VPCs.

You can use a default subnet as you would use any other subnet; you can add custom route tables and set network ACLs. You can also specify a default subnet when you launch an EC2 instance.

Default Subnets

The CIDR block for a default VPC is always a /16 netmask, for example, 172.31.0.0/16. This provides up to 65,536 private IP addresses. The netmask for a default subnet is always /20, which provides up to 4,096 addresses per subnet, a few of which are reserved for our use.

By default, a default subnet is a public subnet, because the main route table sends the subnet's traffic that is destined for the Internet to the Internet gateway. You can make a default subnet a private subnet by removing the route from the destination 0.0.0.0/0 to the Internet gateway. However, if you do this, any EC2 instance running in that subnet can't access the Internet.

From time to time, AWS may add a new Availability Zone to a region. In most cases, we'll automatically create a new default subnet in this Availability Zone for your default VPC. However, if you've made any modifications to your default VPC, we do not add a new default subnet. If you want a default subnet for the new Availability Zone, contact AWS Support to create a default subnet for you.

Detecting Your Supported Platforms and Whether You Have a Default VPC

You can launch EC2 instances into a default VPC and use services such as Elastic Load Balancing, Amazon Relational Database Service (Amazon RDS), and Amazon Elastic MapReduce (Amazon EMR) without needing to know anything about Amazon VPC. Your experience with these services is the same whether you are using a default VPC or EC2-Classic. However, you can use the Amazon EC2 console or the command line to determine whether your AWS account supports both platforms and if you have a default VPC.

Detecting Platform Support Using the Console

The Amazon EC2 console indicates which platforms you can launch EC2 instances into, and whether you have a default VPC.

Verify that the region you'll use is selected in the navigation bar. On the Amazon EC2 console dashboard, look for **Supported Platforms** under **Account Attributes**. If there are two values, `EC2` and `VPC`, you can launch instances into either platform. If there is one value, `VPC`, you can launch instances only into EC2-VPC.

For example, the following indicates that the account supports the EC2-VPC platform only, and has a default VPC with the identifier `vpc-1a2b3c4d`.

If you delete your default VPC, the **Default VPC** value displayed is `None`. For more information, see Deleting Your Default VPC (p. 61).

Detecting Platform Support Using the Command Line

The `supported-platforms` attribute indicates which platforms you can launch EC2 instances into. To get the value of this attribute for your account, use one of the following commands:

- describe-account-attributes (AWS CLI)
- ec2-describe-account-attributes (Amazon EC2 CLI)
- Get-EC2AccountAttributes (AWS Tools for Windows PowerShell)

Also, when you list your VPCs using the following commands, we indicate any default VPCs in the output:

- describe-vpcs (AWS CLI)
- ec2-describe-vpcs (Amazon EC2 CLI)
- Get-EC2Vpc (AWS Tools for Windows PowerShell)

Launching an EC2 Instance into Your Default VPC

When you launch an EC2 instance without specifying a subnet, it's automatically launched into a default subnet in your default VPC. By default, we select an Availability Zone for you and launch the instance into the corresponding subnet for that Availability Zone. Alternatively, you can select the Availability Zone for your instance by selecting its corresponding default subnet in the console, or by specifying the subnet or the Availability Zone in the CLI.

Launching an EC2 Instance Using the Console

To launch an EC2 instance into your default VPC

1. Open the Amazon EC2 console.
2. From the console dashboard, click **Launch Instance**.
3. Follow the directions in the wizard. Select an AMI, and choose an instance type. You can accept the default settings for the rest of the wizard by clicking **Review and Launch**. This takes you directly to the **Review Instance Launch** page.
4. Review your settings. In the **Instance Details** section, the default for **Subnet** is **No preference (default subnet in any Availability Zone)**. This means that the instance is launched into the default subnet of the Availability Zone that we select. Alternatively, you can click **Edit instance details** and select the default subnet for a particular Availability Zone.
5. Click **Launch** to choose a key pair and launch the instance.

Launching an EC2 Instance Using the Command Line

You can use one of the following commands to launch an EC2 instance:

- run-instances (AWS CLI)
- ec2-run-instances (Amazon EC2 CLI)

- New-EC2Instance (AWS Tools for Windows PowerShell)

To launch an EC2 instance into your default VPC, use these commands without specifying a subnet or an Availability Zone.

To launch an EC2 instance into a specific default subnet in your default VPC, specify its subnet ID or Availability Zone.

Deleting Your Default VPC

You can delete one or more of your default subnets just as you can delete any other subnet. However, after you've deleted a default subnet, it's gone. Now, you can't launch EC2 instances into that Availability Zone in your default VPC, unless you create a subnet in that Availability Zone and explicitly launch instances into that subnet. If you delete all default subnets for your default VPC, then you must specify a subnet in another VPC when you launch an EC2 instance, because you can't launch instances into EC2-Classic. For more information, see Deleting Your Subnet (p. 54).

If you try to delete your default subnet, the **Delete Subnet** dialog box displays a warning and requires you to acknowledge that you are aware that you are deleting a default subnet.

You can delete a default VPC just as you can delete any other VPC. However, after you've deleted your default VPC, it's gone. Now, you must specify a subnet in another VPC when you launch an EC2 instance, because you can't launch instances into EC2-Classic. If you try to delete your default VPC, the **Delete VPC** dialog box displays a warning and requires you to acknowledge that you are aware that you are deleting a default VPC. For more information, see Deleting Your VPC (p. 50).

If you delete your default VPC and then need to restore it, you can contact AWS Support to create a new default VPC in that region for you. You cannot mark an existing VPC as a default VPC. If you delete a default subnet and then need to restore it, create a new subnet in your default VPC, and then contact AWS Support to mark the subnet as a default subnet. You must provide the following details: your AWS account ID, the region, and the subnet ID. To ensure that your new default subnet behaves as expected, modify the subnet attribute to assign public IP addresses to instances that are launched in that subnet. For more information, see Modifying Your Subnet's Public IP Addressing Behavior (p. 117). You can only have one default subnet per Availability Zone. You cannot create a default subnet in a nondefault VPC.

Security in Your VPC

Amazon VPC provides two features that you can use to increase security for your VPC:

- Security groups—Act as a firewall for associated Amazon EC2 instances, controlling both inbound and outbound traffic at the instance level
- Network access control lists (ACLs)—Act as a firewall for associated subnets, controlling both inbound and outbound traffic at the subnet level

When you launch an instance in a VPC, you can associate one or more security groups that you've created. Each instance in your VPC could belong to a different set of security groups. If you don't specify a security group when you launch an instance, the instance automatically belongs to the default security group for the VPC. For more information about security groups, see Security Groups for Your VPC (p. 64)

You can secure your VPC instances using only security groups; however, you can add network ACLs as a second layer of defense. For more information about network ACLs, see Network ACLs (p. 70).

You can use AWS Identity and Access Management to control who in your organization has permission to create and manage security groups and network ACLs. For example, you can give only your network administrators that permission, but not personnel who only need to launch instances. For more information, see Controlling Access to Amazon VPC Resources (p. 87).

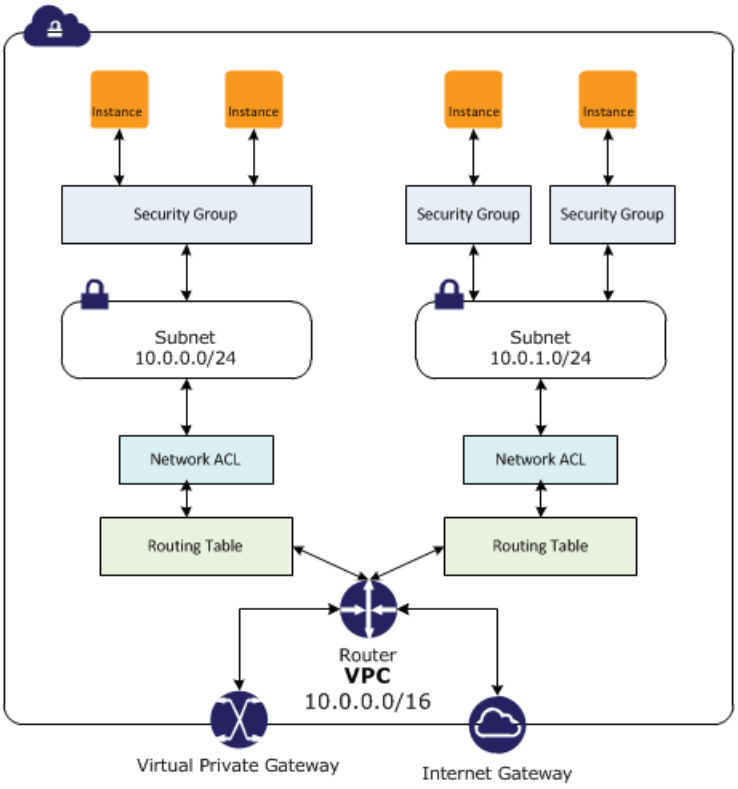
Amazon security groups and network ACLs don't filter traffic to or from link-local addresses (`169.254.0.0/16`) or AWS reserved addresses (the first four IP addresses and the last one in each subnet). These addresses support the services: Domain Name Services (DNS), Dynamic Host Configuration Protocol (DHCP), Amazon EC2 instance metadata, Key Management Server (KMS—license management for Windows instances), and routing in the subnet. You can implement additional firewall solutions in your instances to block network communication with link-local addresses.

Comparison of Security Groups and Network ACLs

The following table summarizes the basic differences between security groups and network ACLs.

Security Group	Network ACL
Operates at the instance level (first layer of defense)	Operates at the subnet level (second layer of defense)
Supports allow rules only	Supports allow rules and deny rules
Is stateful: Return traffic is automatically allowed, regardless of any rules	Is stateless: Return traffic must be explicitly allowed by rules
We evaluate all rules before deciding whether to allow traffic	We process rules in number order when deciding whether to allow traffic
Applies to an instance only if someone specifies the security group when launching the instance, or associates the security group with the instance later on	Automatically applies to all instances in the subnets it's associated with (backup layer of defense, so you don't have to rely on someone specifying the security group)

The following diagram illustrates the layers of security provided by security groups and network ACLs. For example, traffic from an Internet gateway is routed to the appropriate subnet using the routes in the routing table. The rules of the network ACL associated with the subnet control which traffic is allowed to the subnet. The rules of the security group associated with an instance control which traffic is allowed to the instance.



Security Groups for Your VPC

A *security group* acts as a virtual firewall for your instance to control inbound and outbound traffic. When you launch an instance in a VPC, you can assign the instance to up to five security groups. Security groups act at the instance level, not the subnet level. Therefore, each instance in a subnet in your VPC could be assigned to a different set of security groups. If you don't specify a particular group at launch time, the instance is automatically assigned to the default security group for the VPC.

For each security group, you add *rules* that control the inbound traffic to instances, and a separate set of rules that control the outbound traffic. This section describes the basics things you need to know about security groups for your VPC and their rules.

You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC. For more information about the differences between security groups and network ACLs, see Comparison of Security Groups and Network ACLs (p. 62).

Topics

- Security Group Basics (p. 64)
- Default Security Group for Your VPC (p. 64)
- Security Group Rules (p. 65)
- Differences Between Security Groups for EC2-Classic and EC2-VPC (p. 66)
- Working with Security Groups (p. 67)
- API and CLI Overview (p. 69)

Security Group Basics

The following are the basic characteristics of security groups for your VPC:

- You can create up to 100 security groups per VPC. You can add up to 50 rules to each security group. If you need to apply more than 50 rules to an instance, you can associate up to 5 security groups with each network interface. For more information about network interfaces, see Elastic Network Interfaces (ENI).
- You can specify allow rules, but not deny rules.
- You can specify separate rules for inbound and outbound traffic.
- By default, no inbound traffic is allowed until you add inbound rules to the security group.
- By default, an outbound rule allows all outbound traffic. You can remove the rule and add outbound rules that allow specific outbound traffic only.
- Security groups are stateful — responses to allowed inbound traffic are allowed to flow outbound regardless of outbound rules, and vice versa.
- Instances associated with a security group can't talk to each other unless you add rules allowing it (exception: the default security group has these rules by default).
- After you launch an instance, you can change which security groups the instance is associated with.

For information about increasing the limits related to security groups, see Amazon VPC Limits (p. 194).

Default Security Group for Your VPC

Your VPC automatically comes with a default security group. Each EC2 instance that you launch in your VPC is automatically associated with the default security group if you don't specify a different security group when you launch the instance.

The following table describes the default rules for a default security group.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
The security group ID (sg-*xxxxxxxx*)	All	All	Allow inbound traffic from instances assigned to the same security group
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	All	All	Allow all outbound traffic

You can change the rules for the default security group.

You can't delete a default security group. If you try to delete the default security group, you'll get the following error: `Client.CannotDelete: the specified group: "sg-51530134" name: "default" cannot be deleted by a user`.

Security Group Rules

You can add or remove rules for a security group (also referred to as *authorizing* or *revoking* inbound or outbound access). A rule applies either to inbound traffic (ingress) or outbound traffic (egress). You can grant access to a specific CIDR range, or to another security group in your VPC.

The following are the basic parts of a security group rule:

- (Inbound rules only) The source of the traffic (CIDR range or security group) and the destination port or port range
- (Outbound rules only) The destination for the traffic (CIDR range or security group) and the destination port or port range
- Any protocol that has a standard protocol number (for a list, see Protocol Numbers)

  If you specify ICMP as the protocol, you can specify any or all of the ICMP types and codes

When you specify a security group as the source for a rule, this allows instances associated with the source security group to access instances in the security group. (Note that this does not add rules from the source security group to this security group.)

When you add or remove rules, they are automatically applied to all instances associated with the security group.

Some systems for setting up firewalls let you filter on source ports. Security groups let you filter only on destination ports.

The following table describes example rules for a security group for web servers. The web servers can receive HTTP and HTTPS traffic, and send SQL or MySQL traffic to a database server.

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow inbound HTTP access from anywhere

0.0.0.0/0	TCP	443	Allow inbound HTTPS access from anywhere
Your network's public IP address range	TCP	22	Allow inbound SSH access to Linux instances from your network (over the Internet gateway)
Your network's public IP address range	TCP	3389	Allow inbound RDP access to Windows instances from your network (over the Internet gateway)
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
The ID of the security group for your database servers	TCP	1433	Allow outbound Microsoft SQL Server access to instances in the specified security group
The ID of the security group for your MySQL database servers	TCP	3306	Allow outbound MySQL access to instances in the specified security group

For step-by-step directions for creating security groups for web servers and database servers, see Recommended Security Groups (p. 35). For more information about creating security group rules to ensure that Path MTU Discovery can function correctly, see Path MTU Discovery in the *Amazon EC2 User Guide*.

Differences Between Security Groups for EC2-Classic and EC2-VPC

If you're already an Amazon EC2 user, you're probably familiar with security groups. However, you can't use the security groups that you've created for use with EC2-Classic with instances in your VPC. You must create security groups specifically for use with instances in your VPC. The rules you create for use with a security group for a VPC can't reference a security group for EC2-Classic, and vice versa.

The following table summarizes the differences between security groups for use with EC2-Classic and those for use with EC2-VPC.

EC2-Classic	EC2-VPC
You can create up to 500 security groups per region.	You can create up to 100 security groups per VPC.
You can add up to 100 rules to a security group.	You can add up to 50 rules to a security group.
You can add rules for inbound traffic only.	You can add rules for inbound and outbound traffic.
You can assign up to 500 security groups to an instance.	You can assign up to 5 security groups to a network interface.
You can reference security groups from other AWS accounts.	You can reference security groups for your VPC only.
After you launch an instance, you can't change the security groups assigned to it.	You can change the security groups assigned to an instance after it's launched.

EC2-Classic	EC2-VPC
When you add a rule to a security group, you don't have to specify a protocol, and only TCP, UDP, or ICMP are available.	When you add a rule to a security group, you must specify a protocol, and it can be any protocol with a standard protocol number, or all protocols (see Protocol Numbers).
When you add a rule to a security group, you must specify port numbers (for TCP or UDP).	When you add a rule to a security group, you can specify port numbers only if the rule is for TCP or UDP, and you can specify all port numbers.

Working with Security Groups

This section shows you how to work with security groups using the AWS Management Console.

Topics

- Modifying the Default Security Group (p. 67)
- Creating a Security Group (p. 67)
- Adding and Removing Rules (p. 67)
- Changing an Instance's Security Groups (p. 68)
- Deleting a Security Group (p. 68)
- Deleting the 2009-07-15-default Security Group (p. 69)

Modifying the Default Security Group

Your VPC includes a default security group whose initial rules are to deny all inbound traffic, allow all outbound traffic, and allow all traffic between instances in the group. You can't delete this group; however, you can change the group's rules. The procedure is the same as modifying any other security group. For more information, see Adding and Removing Rules (p. 67).

Creating a Security Group

Although you can use the default security group for your instances, you might want to create your own groups to reflect the different roles that instances play in your system.

To create a security group

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Security Groups**.
3. Click the **Create Security Group** button.
4. Enter a name of the security group (for example, `my-security-group`) and provide a description. Select the ID of your VPC from the **VPC** menu, and then click **Yes, Create**.

By default, new security groups start with only an outbound rule that allows all traffic to leave the instances. You must add rules to enable any inbound traffic or to restrict the outbound traffic.

Adding and Removing Rules

When you add or remove a rule, any instances already assigned to the security group are subject to the change. You can't modify rules; you can only add and delete rules.

Several of the scenarios presented in this guide include instructions for adding rules to security groups. For an example, see Recommended Security Groups (p. 35).

To add a rule

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Security Groups**.
3. Select the security group to update. The details pane displays the details for the security group, plus tabs for working with its inbound rules and outbound rules.
4. On the **Inbound Rules** tab, click **Edit**. Select an option for a rule for inbound traffic from the **Type** list, and then fill in the required information. For example, select **HTTP** or **HTTPS** and specify the **Source** as `0.0.0.0/0`. Click **Save** when you are done.
5. You can also allow communication between all instances associated with this security group. On the **Inbound Rules** tab, select **All Traffic** from the **Type** list. Start typing the ID of the security group in the **Source** field; this provides you with a list of security groups. Select the security group from the list, and then click **Save**.
6. If you need to, you can use the **Outbound Rules** tab to add rules for outbound traffic.

To delete a rule

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Security Groups**.
3. Select the security group to update. The details pane displays the details for the security group, plus tabs for working with its inbound rules and outbound rules.
4. Click **Edit**, and then click the **Remove** button for rule you want to delete. Click **Save** when you're done.

Changing an Instance's Security Groups

You can change the security groups that an instance in a VPC is assigned to after the instance is launched. When you make this change, the instance can be either running or stopped.

> **Note**
> This procedure changes the security groups that are associated with the primary network interface (eth0) of the instance. To change the security groups for other network interfaces, see Changing the Security Group of a Network Interface.

To change an instance's security groups

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. Click **Instances** in the navigation pane.
3. Right-click the instance, select **Networking**, and then click **Change Security Groups**.
4. In the **Change Security Groups** dialog box, select one or more security groups from the list, and then click **Assign Security Groups**.

Deleting a Security Group

You can delete a security group only if there are no instances assigned to it (either running or stopped). You can assign the instances to another security group before you delete the security group (see Changing an Instance's Security Groups (p. 68)). You can't delete a default security group.

To delete a security group

1. Open the Amazon VPC console.
2. Click **Security Groups** in the navigation pane.
3. Select the security group, and then click **Delete**.
4. In the **Delete Security Group** dialog box, click **Yes, Delete**.

Deleting the 2009-07-15-default Security Group

Any VPC created using an API version older than 2011-01-01 has the `2009-07-15-default` security group. This security group exists in addition to the regular `default` security group that comes with every VPC. You can't attach an Internet gateway to a VPC that has the `2009-07-15-default` security group. Therefore, you must delete this security group before you can attach an Internet gateway to the VPC.

> **Note**
> If you assigned this security group to any instances, you must assign these instances a different security group before you can delete the security group.

To delete the `2009-07-15-default` security group

1. Ensure that this security group is not assigned to any instances.
   a. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
   b. In the navigation pane, click **Network Interfaces**.
   c. Select the network interface for the instance from the list, and then select **Change Security Groups** from the **Actions** list.
   d. In the **Change Security Groups** dialog box, select a new security group from the list, and then click **Save**.

      > **Tip**
      > When changing an instance's security group, you can select multiple groups from the list. The security groups that you select replace the current security groups for the instance.

   e. Repeat the preceding steps for each instance.
2. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
3. In the navigation pane, click **Security Groups**.
4. Select the `2009-07-15-default` security group, and then click the **Delete** button.
5. In the **Delete Security Group** dialog box, click **Yes, Delete**.

API and CLI Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Create a security group

- create-security-group (AWS CLI)
- ec2-create-group (Amazon EC2 CLI)
- New-EC2SecurityGroup (AWS Tools for Windows PowerShell)

Add a rule to a security group

- authorize-security-group-ingress and authorize-security-group-egress (AWS CLI)
- ec2-authorize (Amazon EC2 CLI)
- Grant-EC2SecurityGroupIngress and Grant-EC2SecurityGroupEgress (AWS Tools for Windows PowerShell)

Describe one or more security groups

- describe-security-groups (AWS CLI)
- ec2-describe-group (Amazon EC2 CLI)
- Get-EC2SecurityGroup (AWS Tools for Windows PowerShell)

Modify the security groups for an instance

- modify-instance-attribute (AWS CLI)
- ec2-modify-instance-attribute (Amazon EC2 CLI)
- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

Remove a rule from a security group

- revoke-security-group-ingress and revoke-security-group-egress(AWS CLI)
- ec2-revoke (Amazon EC2 CLI)
- Revoke-EC2SecurityGroupIngress and Revoke-EC2SecurityGroupEgress (AWS Tools for Windows PowerShell)

Delete a security group

- delete-security-group (AWS CLI)
- ec2-delete-group (Amazon EC2 CLI)
- Remove-EC2SecurityGroup (AWS Tools for Windows PowerShell)

Network ACLs

A *network access control list (ACL)* is an optional layer of security that acts as a firewall for controlling traffic in and out of a subnet. You might set up network ACLs with rules similar to your security groups in order to add an additional layer of security to your VPC. For more information about the differences between security groups and network ACLs, see Comparison of Security Groups and Network ACLs (p. 62).

Topics

- Network ACL Basics (p. 71)
- Network ACL Rules (p. 71)
- Default Network ACL (p. 71)
- Example Custom Network ACL (p. 72)
- Ephemeral Ports (p. 74)
- Working with Network ACLs (p. 74)
- API and Command Overview (p. 131)

Network ACL Basics

The following are the basic things that you need to know about network ACLs:

- A network ACL is a numbered list of rules that we evaluate in order, starting with the lowest numbered rule, to determine whether traffic is allowed in or out of any subnet associated with the network ACL. The highest number that you can use for a rule is 32766. We suggest that you start by creating rules with rule numbers that are multiples of 100, so that you can insert new rules where you need to later on.
- A network ACL has separate inbound and outbound rules, and each rule can either allow or deny traffic.
- Your VPC automatically comes with a modifiable default network ACL; by default, it allows all inbound and outbound traffic.
- You can create custom network ACLs; each custom network ACL starts out closed (permits no traffic) until you add a rule.
- Each subnet must be associated with a network ACL; if you don't explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL.
- Network ACLs are stateless; responses to allowed inbound traffic are subject to the rules for outbound traffic (and vice versa).

For information about the number of network ACLs you can create, see Amazon VPC Limits (p. 194).

Network ACL Rules

You can add or remove rules from the default network ACL, or create additional network ACLs for your VPC. When you add or remove rules from a network ACL, the changes are automatically applied to the subnets it's associated with.

The following are the parts of a network ACL rule:

- Rule number. Rules are evaluated starting with the lowest numbered rule.
- Protocol. You can specify any protocol that has a standard protocol number. For more information, see Protocol Numbers. If you specify ICMP as the protocol, you can specify any or all of the ICMP types and codes.
- [Inbound rules only] The source of the traffic (CIDR range) and the destination (listening) port or port range.
- [Outbound rules only] The destination for the traffic (CIDR range) and the destination port or port range.
- Choice of allow or deny.

Default Network ACL

To help you understand what ACL rules look like, here's what the default network ACL looks like in its initial state. It is configured to allow all traffic to flow in and out of each subnet. Each network ACL includes a rule whose rule number is an asterisk. This rule ensures that if a packet doesn't match any of the other rules, it's denied. You can't modify or remove this rule.

Inbound				
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**
100	0.0.0.0/0	All	All	ALLOW
*	0.0.0.0/0	All	All	DENY

Outbound				
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**
100	0.0.0.0/0	all	all	ALLOW
*	0.0.0.0/0	all	all	DENY

Example Custom Network ACL

The following table shows an example of a custom network ACL. It includes rules that allow HTTP and HTTPS traffic in (inbound rules 100 and 110). There's a corresponding outbound rule that enables responses to that inbound traffic (outbound rule 120, which covers ephemeral ports 49152-65535). For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 74).

The network ACL also includes inbound rules that allow SSH and RDP traffic into the subnet. The outbound rule 120 enables responses to egress the subnet.

The network ACL has outbound rules (100 and 110) that allow outbound HTTP and HTTPS traffic out of the subnet. There's a corresponding inbound rule that enables responses to that outbound traffic (inbound rule 140, which covers ephemeral ports 49152-65535).

> **Note**
> Each network ACL includes a default rule whose rule number is an asterisk. This rule ensures that if a packet doesn't match any of the other rules, it's denied. You can't modify or remove this rule.

Inbound					
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic from anywhere.
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic from anywhere.
120	192.0.2.0/24	TCP	22	ALLOW	Allows inbound SSH traffic from your home network's public IP address range (over the Internet gateway).
130	192.0.2.0/24	TCP	3389	ALLOW	Allows inbound RDP traffic to the web servers from your home network's public IP address range (over the Internet gateway).
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from the Internet (that is, for requests that originate in the subnet). For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 74).

150	0.0.0.0/0	UDP	32768-61000	ALLOW	Allows inbound return UDP traffic. For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 74).
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable).
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet.
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet.
120	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet). For more information about how to select the appropriate ephemeral port range, see Ephemeral Ports (p. 74).
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable).

As a packet comes to the subnet, we evaluate it against the ingress rules of the ACL the subnet is associated with (starting at the top of the list of rules, and moving to the bottom). Let's say the packet is destined for the SSL port (443). The packet doesn't match the first rule evaluated (rule 100). It does match the second rule (110), which allows the packet into the subnet. If the packet had been destined for port 139 (NetBIOS), the first two rules would not have matched, but the * rule ultimately would have denied the packet.

You might want to add a DENY rule in a situation where you legitimately need to open a wide range of ports, but there are certain ports within that range you want to deny. Just make sure to place the DENY rule earlier in the table than the rule that allows the wide range of port traffic.

Important

With Elastic Load Balancing, if the subnet for your back-end instances has a network ACL with an inbound DENY rule for all traffic with a source of `0.0.0.0/0`, then your load balancer can't carry out health checks on the instances. To ensure that health checks can be performed, first remove the DENY rule. Next, add two inbound ALLOW rules for each subnet attached to your load balancer: one for all traffic on the listener port and another for all traffic on the health check port. Finally, add an outbound ALLOW rule for each subnet attached to your load balancer for ephemeral ports 1024 to 65535.

Ephemeral Ports

The example network ACL in the preceding section uses an ephemeral port range of 49152-65535. However, you might want to use a different range for your network ACLs. This section explains why.

The client that initiates the request chooses the ephemeral port range. The range varies depending on the client's operating system. Many Linux kernels (including the Amazon Linux kernel) use ports 32768-61000. Requests originating from Elastic Load Balancing use ports 1024-65535. Windows operating systems through Windows Server 2003 use ports 1025-5000. Windows Server 2008 uses ports 49152-65535. Therefore, if a request comes in to a web server in your VPC from a Windows XP client on the Internet, your network ACL must have an outbound rule to enable traffic destined for ports 1025-5000.

If an EC2 instance in your VPC is the client initiating a request, your network ACL must have an inbound rule to enable traffic destined for the ephemeral ports specific to the type of instance (Amazon Linux, Windows Server 2008, and so on.).

In practice, to cover the different types of clients that might initiate traffic to public-facing instances in your VPC, you need to open ephemeral ports 1024-65535. However, you can also add rules to the ACL to deny traffic on any malicious ports within that range. Make sure to place the DENY rules earlier in the table than the rule that opens the wide range of ephemeral ports.

Working with Network ACLs

This section shows you how to work with network ACLs using the Amazon VPC console.

Topics

- Determining Which Network ACL a Subnet Is Associated With (p. 74)
- Determining Which Subnets Are Associated with a Network ACL (p. 75)
- Creating a Network ACL (p. 75)
- Adding and Deleting Rules (p. 75)
- Associating a Subnet with a Network ACL (p. 76)
- Disassociating a Network ACL from a Subnet (p. 76)
- Changing a Subnet's Network ACL (p. 77)
- Deleting a Network ACL (p. 77)

Determining Which Network ACL a Subnet Is Associated With

To determine which network ACL a subnet is associated with

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Subnets**, and then select the subnet.

   The network ACL associated with the subnet is included in the **Network ACL** tab, along with the network ACL's rules.

Determining Which Subnets Are Associated with a Network ACL

To determine which subnets are associated with a network ACL

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Network ACLs**.

   The console displays your network ACLs. The **Associated With** column indicates the number of associated subnets.
3. Select a network ACL.
4. In the details pane, click the **Subnet Associations** tab to display the subnets associated with the network ACL.

Creating a Network ACL

To create a network ACL

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Network ACLs**.
3. Click the **Create Network ACL** button.
4. In the **Create Network ACL** dialog box, optionally name your network ACL, and then select the ID of your VPC from the **VPC** list, and click **Yes, Create**.

The initial settings for a network ACL block all inbound and outbound traffic. The network ACL has no rules except the * rule present in every ACL.

There are no subnets associated with a new ACL.

Adding and Deleting Rules

When you add or delete a rule from an ACL, any subnets associated with the ACL are subject to the change. You don't have to terminate and relaunch the instances in the subnet; the changes take effect after a short period.

You can't modify rules; you can only add and delete rules. If you need to change the order of a rule in the ACL, you must add a new rule with the new rule number, and then delete the original rule.

To add rules to a network ACL

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Network ACLs**.
3. In the details pane, select either the **Inbound Rules** or **Outbound Rules** tab, depending on the type of rule that you need to add, and then click **Edit**.
4. In **Rule #**, enter a rule number (for example, 100). The rule number must not already be used in the network ACL. We process the rules in order, starting with the lowest number.

   Tip
   We recommend that you leave gaps between the rule numbers (such as 100, 200, 300), rather than using sequential numbers (101, 102, 103). This makes it easier add a new rule where it belongs without having to renumber the existing rules.

5. Select a rule from the **Type** list. For example, to add a rule for HTTP, select the **HTTP** option. To add a rule to allow all TCP traffic, select **All TCP**. For some of these options (for example, HTTP), we fill in the port for you. To use a protocol that's not listed, select **Custom protocol rule**.
6. (Optional) If you're creating a custom protocol rule, select the protocol's number and name from the **Protocol** list. For more information, see IANA List of Protocol Numbers.
7. (Optional) If the protocol you've selected requires a port number, enter the port number or port range separated by a hyphen (for example, 49152-65535).
8. In the **Source** or **Destination** box (depending on whether this is an inbound or outbound rule), enter the CIDR range that the rule applies to.
9. From the **Allow/Deny** list, select **ALLOW** to allow the specified traffic or **DENY** to deny the specified traffic.
10. (Optional) To add another rule, click **Add another rule**, and repeat steps 4 to 9 as required.
11. When you are done, click **Save**.

To delete a rule from a network ACL

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Network ACLs**, and then select the network ACL.
3. In the details pane, select either the **Inbound Rules** or **Outbound Rules** tab, and then click **Edit**. Click the **Remove** button for the rule you want to delete, and then click **Save**.

Associating a Subnet with a Network ACL

To apply the rules of a network ACL to a particular subnet, you must associate the subnet with the network ACL. You can associate a network ACL with multiple subnets; however, a subnet can be associated with only one network ACL. Any subnet not associated with a particular ACL is associated with the default network ACL by default.

To associate a subnet with a network ACL

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Network ACLs**, and then select the network ACL.
3. In the details pane, on the **Subnet Associations** tab, click **Edit**. Select the **Associate** check box for the subnet to associate with the table, and then click **Save**.

Disassociating a Network ACL from a Subnet

You might want to disassociate a subnet from its network ACL. For example, you might have a subnet that is associated with a custom network ACL, and you instead want it associated with the default network ACL. By disassociating the subnet from the custom network ACL, the subnet becomes associated with the default network ACL.

To disassociate a subnet from a network ACL

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Network ACLs**, and then select the network ACL.
3. In the details pane, click the **Subnet Associations** tab.
4. Click **Edit**, and then deselect the **Associate** check box for the subnet. Click **Save**.

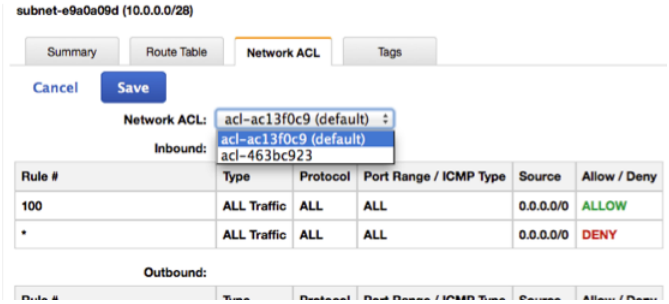
Changing a Subnet's Network ACL

You can change which network ACL a subnet is associated with. For example, when you create a subnet, it is initially associated with the default network ACL. You might want to instead associate it with a custom network ACL that you've created.

After changing a subnet's network ACL, you don't have to terminate and relaunch the instances in the subnet; the changes take effect after a short period.

To change a subnet's network ACL association

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Subnets**, and then select the subnet.
3. Click the **Network ACL** tab, and the click **Edit**.
4. Select the network ACL to associate the subnet with from the **Network ACL** list, and then click **Save**.



Deleting a Network ACL

You can delete a network ACL only if there are no subnets associated with it. You can't delete the default network ACL.

To delete a network ACL

1. Open the Amazon VPC console.
2. Click **Network ACLs** in the navigation pane.
3. Select the network ACL, and then click the **Delete** button.
4. In the **Delete Network ACL** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Create a network ACL for your VPC

- create-network-acl (AWS CLI)
- ec2-create-network-acl (Amazon EC2 CLI)
- New-EC2NetworkAcl (AWS Tools for Windows PowerShell)

Describe one or more of your network ACLs

- describe-network-acls (AWS CLI)
- ec2-describe-network-acls (Amazon EC2 CLI)
- Get-EC2NetworkAcl (AWS Tools for Windows PowerShell)

Add a rule to a network ACL

- create-network-acl-entry (AWS CLI)
- ec2-create-network-acl-entry (Amazon EC2 CLI)
- New-EC2NetworkAclEntry (AWS Tools for Windows PowerShell)

Delete a rule from a network ACL

- delete-network-acl-entry (AWS CLI)
- ec2-delete-network-acl-entry (Amazon EC2 CLI)
- Remove-EC2NetworkAclEntry (AWS Tools for Windows PowerShell)

Replace an existing rule in a network ACL

- replace-network-acl-entry (AWS CLI)
- ec2-replace-network-acl-entry (Amazon EC2 CLI)
- Set-EC2NetworkAclEntry (AWS Tools for Windows PowerShell)

Replace a network ACL association

- replace-network-acl-association (AWS CLI)
- ec2-replace-network-acl-association (Amazon EC2 CLI)
- Set-EC2NetworkAclAssociation (AWS Tools for Windows PowerShell)

Delete a network ACL

- delete-network-acl (AWS CLI)
- ec2-delete-network-acl (Amazon EC2 CLI)
- Remove-EC2NetworkAcl (AWS Tools for Windows PowerShell)

Recommended Network ACL Rules for Your VPC

The VPC wizard helps you implement common scenarios for Amazon VPC. If you implement these scenarios as described in the documentation, you'll use the default network access control list (ACL), which allows all inbound and outbound traffic. If you need an additional layer of security, you can create a network ACL and add rules. We recommend the following rules for each scenario.

Topics

- Recommended Rules for Scenario 1 (p. 79)
- Recommended Rules for Scenario 2 (p. 80)
- Recommended Rules for Scenario 3 (p. 83)
- Recommended Rules for Scenario 4 (p. 86)

For more information about network ACLs and how to use them, see Network ACLs (p. 70).

Important
We use the ephemeral port range 49152-65535. You can select a different range. For more information, see Ephemeral Ports (p. 74).

Recommended Rules for Scenario 1

Scenario 1 is a single subnet with instances that can receive and send Internet traffic. For more information, see Scenario 1: VPC with a Single Public Subnet (p. 17).

The following table shows the rules we recommended. They block all traffic except that which is explicitly required.

Inbound					
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic from anywhere
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic from anywhere
120	Public IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic from your home network (over the Internet gateway)
130	Public IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic from your home network (over the Internet gateway)
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet

110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Recommended Rules for Scenario 2

Scenario 2 is a public subnet with instances that can receive and send Internet traffic, and a private subnet that can't receive traffic directly from the Internet. However, it can initiate traffic to the Internet (and receive responses) through a NAT instance in the public subnet. For more information, see Scenario 2: VPC with Public and Private Subnets (NAT) (p. 22).

For this scenario you have a network ACL for the public subnet, and a separate one for the private subnet. The following table shows the rules we recommend for each ACL. They block all traffic except that which is explicitly required. They mostly mimic the security group rules for the scenario.

ACL Rules for the Public Subnet

Inbound					
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic from anywhere
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic from anywhere
120	Public IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic from your home network (over the Internet gateway)
130	Public IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic from your home network (over the Internet gateway)

140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	10.0.1.0/24	TCP	1433	ALLOW	Allows outbound MS SQL access to database servers in the private subnet
130	10.0.1.0/24	TCP	3306	ALLOW	Allows outbound MySQL access to database servers in the private subnet
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
150	10.0.1.0/24	TCP	22	ALLOW	Allows outbound SSH access to instances in your private subnet (from the SSH bastion)
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

ACL Rules for the Private Subnet

Inbound

Rule #	Source IP	Protocol	Port	Allow/Deny	Comments
100	10.0.0.0/24	TCP	1433	ALLOW	Allows web servers in the public subnet to read and write to MS SQL servers in the private subnet
110	10.0.0.0/24	TCP	3306	ALLOW	Allows web servers in the public subnet to read and write to MySQL servers in the private subnet
120	10.0.0.0/24	TCP	22	ALLOW	Allows inbound SSH traffic from the SSH bastion in the public subnet
130	10.0.0.0/24	TCP	3389	ALLOW	Allows inbound RDP traffic from the Microsoft Terminal Services gateway in the public subnet
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from NAT instance in the public subnet for requests originating in the private subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet

120	10.0.0.0/24	TCP	49152-65535	ALLOW	Allows outbound responses to the public subnet (for example, responses to web servers in the public subnet that are communicating with DB Servers in the private subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Recommended Rules for Scenario 3

Scenario 3 is a public subnet with instances that can receive and send Internet traffic, and a VPN-only subnet with instances that can communicate only with your home network over the VPN connection. For more information, see Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 32).

For this scenario you have a network ACL for the public subnet, and a separate one for the VPN-only subnet. The following table shows the rules we recommend for each ACL. They block all traffic except that which is explicitly required.

ACL Rules for the Public Subnet

Inbound					
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows inbound HTTP traffic to the web servers from anywhere
110	0.0.0.0/0	TCP	443	ALLOW	Allows inbound HTTPS traffic to the web servers from anywhere
120	Public IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic to the web servers from your home network (over the Internet gateway)
130	Public IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic to the web servers from your home network (over the Internet gateway)

140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	0.0.0.0/0	TCP	80	ALLOW	Allows outbound HTTP traffic from the subnet to the Internet
110	0.0.0.0/0	TCP	443	ALLOW	Allows outbound HTTPS traffic from the subnet to the Internet
120	10.0.1.0/24	TCP	1433	ALLOW	Allows outbound MS SQL access to database servers in the VPN-only subnet
130	10.0.1.0/24	TCP	3306	ALLOW	Allows outbound MySQL access to database servers in the VPN-only subnet
140	0.0.0.0/0	TCP	49152-65535	ALLOW	Allows outbound responses to clients on the Internet (for example, serving web pages to people visiting the web servers in the subnet) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

ACL Settings for the VPN-Only Subnet

Inbound					
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**

100	10.0.0.0/24	TCP	1433	ALLOW	Allows web servers in the public subnet to read and write to MS SQL servers in the VPN-only subnet
110	10.0.0.0/24	TCP	3306	ALLOW	Allows web servers in the public subnet to read and write to MySQL servers in the VPN-only subnet
120	Private IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic from the home network (over the virtual private gateway)
130	Private IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic from the home network (over the virtual private gateway)
140	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows inbound return traffic from clients in the home network (over the virtual private gateway) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	Private IP address range of your home network	All	All	ALLOW	Allows all outbound traffic from the subnet to your home network (over the virtual private gateway)
110	10.0.0.0/24	TCP	49152-65535	ALLOW	Allows outbound responses to the web servers in the public subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.

120	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows outbound responses to clients in the home network (over the virtual private gateway) See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Recommended Rules for Scenario 4

Scenario 4 is a single subnet with instances that can communicate only with your home network over a VPN connection. For a more information, see Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access (p. 41).

The following table shows the rules we recommended. They block all traffic except that which is explicitly required.

Inbound					
Rule #	**Source IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**
100	Private IP address range of your home network	TCP	22	ALLOW	Allows inbound SSH traffic to the subnet from your home network
110	Private IP address range of your home network	TCP	3389	ALLOW	Allows inbound RDP traffic to the subnet from your home network
120	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows inbound return traffic from requests originating in the subnet See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all inbound traffic not already handled by a preceding rule (not modifiable)
Outbound					
Rule #	**Dest IP**	**Protocol**	**Port**	**Allow/Deny**	**Comments**

100	Private IP address range of your home network	All	All	ALLOW	Allows all outbound traffic from the subnet to your home network
120	Private IP address range of your home network	TCP	49152-65535	ALLOW	Allows outbound responses to clients in the home network See the important note at the beginning of this topic about specifying the correct ephemeral ports.
*	0.0.0.0/0	all	all	DENY	Denies all outbound traffic not already handled by a preceding rule (not modifiable)

Controlling Access to Amazon VPC Resources

Your security credentials identify you to services in AWS and grant you unlimited use of your AWS resources, such as your Amazon VPC resources. You can use AWS Identity and Access Management (IAM) to allow other users, services, and applications to use your Amazon VPC resources without sharing your security credentials. You can choose to allow full use or limited use of your resources by granting users permission to use specific Amazon EC2 API actions. Some API actions support resource-level permissions, which allow you to control the specific resources that users can create or modify.

Important
Currently, not all Amazon EC2 API actions support resource-level permissions. If an Amazon EC2 API action does not support resource-level permissions, you can grant users permission to use the action, but you have to specify a * for the resource element of your policy statement. For an example of how to do this, see the following example policy: 1. Managing a VPC (p. 89) We'll add support for additional API actions and ARNs for additional Amazon EC2 resources later. For information about which ARNs you can use with which Amazon EC2 API actions, as well as supported condition keys for each ARN, see Supported Resources and Conditions for Amazon EC2 API Actions in the *Amazon EC2 User Guide for Linux Instances*.

For more information about creating IAM policies for Amazon EC2, supported resources for EC2 API actions, as well example policies for Amazon EC2, see IAM Policies for Amazon EC2 in the *Amazon EC2 User Guide for Linux Instances*.

Topics

- Example Policies for a CLI or SDK (p. 87)
- Example Policies for the Console (p. 97)

Example Policies for a CLI or SDK

The following examples show policy statements that you can use to control the permissions that IAM users have to Amazon VPC. These examples are designed for users that use the AWS CLI, the Amazon EC2 CLI, or an AWS SDK.

- 1. Managing a VPC (p. 89)

- 2. Read-Only Policy for Amazon VPC (p. 90)
- 3. Custom Policy for Amazon VPC (p. 90)
- 4. Launching instances into a specific subnet (p. 91)
- 5. Launching instances into a specific VPC (p. 92)
- 6. Managing security groups in a VPC (p. 93)
- 7. Creating and managing VPC peering connections (p. 94)
- 8. Creating and managing VPC endpoints (p. 97)

For example policies for working with ClassicLink, see Example Policies for CLI or SDK in the *Amazon EC2 User Guide for Linux Instances*.

Example 1. Managing a VPC

The following policy grants users permission to create and manage your VPC. You might attach this policy to a group of network administrators. The `Action` element specifies the API actions related to VPCs, subnets, Internet gateways, customer gateways, virtual private gateways, VPN connections, route tables, Elastic IP addresses, security groups, network ACLs, and DHCP options sets. The policy also allows the group to run, stop, start, and terminate instances. It also allows the group to list Amazon EC2 resources.

The policy uses wildcards to specify all actions for each type of object (for example, `*SecurityGroup*`). Alternatively, you could list each action explicitly. If you use the wildcards, be aware that if we add new actions whose names include any of the wildcarded strings in the policy, the policy would automatically grant the group access to those new actions.

The `Resource` element uses a wildcard to indicate that users can specify all resources with these API actions. The * wildcard is also necessary in cases where the API action does not support resource-level permissions.

```
{
    "Version": "2012-10-17",
    "Statement":[{
    "Effect":"Allow",
    "Action":["ec2:*Vpc*",
              "ec2:*Subnet*",
              "ec2:*Gateway*",
              "ec2:*Vpn*",
              "ec2:*Route*",
              "ec2:*Address*",
              "ec2:*SecurityGroup*",
              "ec2:*NetworkAcl*",
              "ec2:*DhcpOptions*",
              "ec2:RunInstances",
              "ec2:StopInstances",
              "ec2:StartInstances",
              "ec2:TerminateInstances",
              "ec2:Describe*"],
    "Resource":"*"
    }
  ]
  }
```

Example 2. Read-Only Policy for Amazon VPC

The following policy grants users permission to list your VPCs and their components. They can't create, update, or delete them.

```
{
    "Version": "2012-10-17",
    "Statement":[{
    "Effect":"Allow",
    "Action":["ec2:DescribeVpcs",
              "ec2:DescribeSubnets",
              "ec2:DescribeInternetGateways",
              "ec2:DescribeCustomerGateways",
              "ec2:DescribeVpnGateways",
              "ec2:DescribeVpnConnections",
              "ec2:DescribeRouteTables",
              "ec2:DescribeAddresses",
              "ec2:DescribeSecurityGroups",
              "ec2:DescribeNetworkAcls",
              "ec2:DescribeDhcpOptions",
              "ec2:DescribeTags",
              "ec2:DescribeInstances"],
    "Resource":"*"
    }
  ]
}
```

Example 3. Custom Policy for Amazon VPC

The following policy grants users permission to launch instances, stop instances, start instances, terminate instances, and describe the available resources for Amazon EC2 and Amazon VPC.

The second statement in the policy protects against any other policy that might grant the user access to a wider range of API actions by explicitly denying permissions.

```
{
  "Version": "2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Action":["ec2:RunInstances",
              "ec2:StopInstances",
              "ec2:StartInstances",
              "ec2:TerminateInstances",
              "ec2:Describe*"],
     "Resource":"*"
     },
     {
     "Effect":"Deny",
     "NotAction":["ec2:RunInstances",
                  "ec2:StopInstances",
                  "ec2:StartInstances",
                  "ec2:TerminateInstances",
                  "ec2:Describe*"],
     "Resource":"*"
     }
  ]
}
```

Example 4. Launching instances into a specific subnet

The following policy grants users permission to launch instances into a specific subnet, and to use a specific security group in the request. The policy does this by specifying the ARN for `subnet-1a2b3c4d`, and the ARN for `sg-123abc123`. If users attempt to launch an instance into a different subnet or using a different security group, the request will fail (unless another policy or statement grants users permission to do so).

The policy also grants permission to use the network interface resource. When launching into a subnet, the `RunInstances` request creates a primary network interface by default, so the user needs permission to create this resource when launching the instance.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:region::image/ami-*",
        "arn:aws:ec2:region:account:instance/*",
        "arn:aws:ec2:region:account:subnet/subnet-1a2b3c4d",
        "arn:aws:ec2:region:account:network-interface/*",
        "arn:aws:ec2:region:account:volume/*",
        "arn:aws:ec2:region:account:key-pair/*",
        "arn:aws:ec2:region:account:security-group/sg-123abc123"
      ]
    }
   ]
}
```

Example 5. Launching instances into a specific VPC

The following policy grants users permission to launch instances into any subnet within a specific VPC. The policy does this by applying a condition key (`ec2:Vpc`) to the subnet resource.

The policy also grants users permission to launch instances using only AMIs that have the tag "`department=dev`".

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "arn:aws:ec2:region:account:subnet/*",
        "Condition": {
         "StringEquals": {
            "ec2:Vpc": "arn:aws:ec2:region:account:vpc/vpc-1a2b3c4d"
            }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "arn:aws:ec2:region::image/ami-*",
      "Condition": {
         "StringEquals": {
            "ec2:ResourceTag/department": "dev"
            }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:region:account:instance/*",
         "arn:aws:ec2:region:account:volume/*",
         "arn:aws:ec2:region:account:network-interface/*",
         "arn:aws:ec2:region:account:key-pair/*",
         "arn:aws:ec2:region:account:security-group/*"
         ]
      }
   ]
}
```

Example 6. Managing security groups in a VPC

The following policy grants users permission to create and delete inbound and outbound rules for any security group within a specific VPC. The policy does this by applying a condition key (`ec2:Vpc`) to the security group resource for the `Authorize` and `Revoke` actions.

The second statement grants users permission to describe all security groups. This is necessary in order for users to be able to modify security group rules using the CLI.

```
{
"Version": "2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Action": [
       "ec2:AuthorizeSecurityGroupIngress",
       "ec2:AuthorizeSecurityGroupEgress",
       "ec2:RevokeSecurityGroupIngress",
       "ec2:RevokeSecurityGroupEgress"],
     "Resource": "arn:aws:ec2:region:account:security-group/*",
      "Condition": {
        "StringEquals": {
          "ec2:Vpc": "arn:aws:ec2:region:account:vpc/vpc-1a2b3c4d"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:DescribeSecurityGroups",
      "Resource": "*"
    }
  ]
}
```

Example 7. Creating and managing VPC peering connections

The following are examples of policies you can use to manage the creation and modification of VPC peering connections.

a. Create a VPC peering connection

The following policy allows users to create VPC peering connection requests using only VPCs that are tagged with `Purpose=Peering`. The first statement applies a condition key (`ec2:ResourceTag`) to the VPC resource. Note that the VPC resource for the `CreateVpcPeeringConnection` action is always the requester VPC.

The second statement grants users permissions to create the VPC peering connection resource, and therefore uses the * wildcard in place of a specific resource ID.

```
{
"Version": "2012-10-17",
"Statement":[{
 "Effect":"Allow",
 "Action": "ec2:CreateVpcPeeringConnection",
 "Resource": "arn:aws:ec2:region:account:vpc/*",
  "Condition": {
    "StringEquals": {
     "ec2:ResourceTag/Purpose": "Peering"
    }
   }
  },
  {
  "Effect": "Allow",
  "Action": "ec2:CreateVpcPeeringConnection",
  "Resource": "arn:aws:ec2:region:account:vpc-peering-connection/*"
  }
 ]
}
```

The following policy allows users in AWS account 333333333333 to create VPC peering connections using any VPC in the cn-north-1 region, but only if the VPC that will be accepting the peering connection is a specific VPC (`vpc-aaa111bb`) in a specific account (777788889999).

```
{
"Version": "2012-10-17",
"Statement": [{
 "Effect":"Allow",
 "Action": "ec2:CreateVpcPeeringConnection",
 "Resource": "arn:aws:ec2:cn-north-1:333333333333:vpc/*"
 },
 {
  "Effect": "Allow",
  "Action": "ec2:CreateVpcPeeringConnection",
  "Resource": "arn:aws:ec2:region:333333333333:vpc-peering-connection/*",
    "Condition": {
     "ArnEquals": {
       "ec2:AccepterVpc": "arn:aws:ec2:region:777788889999:vpc/vpc-aaa111bb"
     }
    }
   }
```

```
 ]
}
```

b. Accept a VPC peering connection

The following policy allows users to accept VPC peering connection requests from AWS account 444455556666 only. This helps to prevent users from accepting VPC peering connection requests from unknown accounts. The first statement uses the `ec2:RequesterVpc` condition key to enforce this.

The policy also grants users permissions to accept VPC peering requests only when your VPC has the tag `Purpose=Peering`.

```
{
"Version": "2012-10-17",
"Statement":[{
 "Effect":"Allow",
 "Action": "ec2:AcceptVpcPeeringConnection",
 "Resource": "arn:aws:ec2:region:account:vpc-peering-connection/*",
  "Condition": {
   "ArnEquals": {
    "ec2:RequesterVpc": "arn:aws:ec2:region:444455556666:vpc/*"
   }
  }
 },
 {
 "Effect": "Allow",
 "Action": "ec2:AcceptVpcPeeringConnection",
 "Resource": "arn:aws:ec2:region:account:vpc/*",
  "Condition": {
   "StringEquals": {
    "ec2:ResourceTag/Purpose": "Peering"
    }
   }
  }
 ]
}
```

c. Deleting a VPC peering connection

The following policy allows users in account 444455556666 to delete any VPC peering connection, except those that use the specified VPC `vpc-1a2b3c4d`, which is in the same account. The policy specifies both the `ec2:AccepterVpc` and `ec2:RequesterVpc` condition keys, as the VPC may have been the requester VPC or the peer VPC in the original VPC peering connection request.

```
{
"Version": "2012-10-17",
"Statement": [{
  "Effect":"Allow",
  "Action": "ec2:DeleteVpcPeeringConnection",
  "Resource": "arn:aws:ec2:region:444455556666:vpc-peering-connection/*",
   "Condition": {
    "ArnNotEquals": {
     "ec2:AccepterVpc": "arn:aws:ec2:region:444455556666:vpc/vpc-1a2b3c4d",
     "ec2:RequesterVpc": "arn:aws:ec2:region:444455556666:vpc/vpc-1a2b3c4d"
    }
   }
  }
 ]
}
```

d. Working within a specific account

The following policy allows users to work with VPC peering connections entirely within a specific account. Users can view, create, accept, reject, and delete VPC peering connections, provided they are all within AWS account 333333333333.

The first statement allows users to view all VPC peering connections. The `Resource` element requires a * wildcard in this case, as this API action (`DescribeVpcPeeringConnections`) currently does not support resource-level permissions.

The second statement allows users to create VPC peering connections, and allows access to all VPCs in account 333333333333 in order to do so.

The third statement uses a * wildcard as part of the `Action` element to allow all VPC peering connection actions. The condition keys ensure that the actions can only be performed on VPC peering connections with VPCs that are part of account 333333333333. For example, a user is not allowed to delete a VPC peering connection if either the accepter or requester VPC is in a different account. A user cannot create a VPC peering connection with a VPC in a different account.

```
{
"Version": "2012-10-17",
"Statement": [{
 "Effect": "Allow",
 "Action": "ec2:DescribeVpcPeeringConnections",
 "Resource": "*"
  },
  {
  "Effect": "Allow",
  "Action": ["ec2:CreateVpcPeeringConnection","ec2:AcceptVpcPeeringConnection"],

  "Resource": "arn:aws:ec2:*:333333333333:vpc/*"
  },
  {
  "Effect": "Allow",
  "Action": "ec2:*VpcPeeringConnection",
  "Resource": "arn:aws:ec2:*:333333333333:vpc-peering-connection/*",
  "Condition": {
   "ArnEquals": {
    "ec2:AccepterVpc": "arn:aws:ec2:*:333333333333:vpc/*",
    "ec2:RequesterVpc": "arn:aws:ec2:*:333333333333:vpc/*"
    }
   }
  }
 ]
}
```

Example 8. Creating and managing VPC endpoints

The following policy grants users permission to create, modify, view, and delete VPC endpoints. None of the `ec2:*VpcEndpoint*` actions support resource-level permissions, so you have to use the * wildcard for the `Resource` element to allow users to work with all resources.

```
{
    "Version": "2012-10-17",
    "Statement":[{
    "Effect":"Allow",
    "Action":"ec2:*VpcEndpoint*",
    "Resource":"*"
    }
  ]
}
```

Example Policies for the Console

You can use IAM policies to grant users permissions to view and work with specific resources in the Amazon VPC console. You can use the example policies in the previous section; however, they are designed for requests that are made with the AWS CLI, the Amazon EC2 CLI, or an AWS SDK. The console uses additional API actions for its features, so these policies may not work as expected.

This section demonstrates policies that enable users to work with specific parts of the VPC console.

- 1. Using the VPC wizard (p. 99)
- 2. Managing a VPC (p. 103)
- 3. Managing security groups (p. 105)

- 4. Creating a VPC peering connection (p. 106)

Example 1. Using the VPC wizard

You can use the VPC wizard in the Amazon VPC console to create and set up and configure a VPC for you, so that it's ready for you to use. The wizard provides different configuration options, depending on your requirements. For more information about using the VPC wizard to create a VPC, see VPC Wizard Scenarios for Amazon VPC (p. 17).

To enable users to use the VPC wizard, you must grant them permission to create and modify the resources that form part of the selected configuration. The following example policies show the actions that are required for each of the wizard configuration options.

> **Note**
> If the VPC wizard fails at any point, it attempts to detach and delete the resources that it's created. If you do not grant users permissions to use these actions, then those resources remain in your account.

Option 1: VPC with a single public subnet

The first VPC wizard configuration option creates a VPC with a single subnet. In your IAM policy, you must grant users permission to use the following actions so they can successfully use this wizard option:

- `ec2:CreateVpc`, `ec2:CreateSubnet`, `ec2:CreateRouteTable`, and `ec2:CreateInternetGateway`: To create a VPC, a subnet, a custom route table, and an Internet gateway.
- `ec2:DescribeAvailabilityZones`: To display the section of the wizard with the **Availability Zone** list and the CIDR block field for the subnet. Even if users intend to leave the default settings, they will not be able to create a VPC unless those options are displayed.
- `ec2:AttachInternetGateway`: To attach the Internet gateway to the VPC.
- `ec2:CreateRoute`: To create a route in the custom route table. The route points traffic to the Internet gateway.
- `ec2:AssociateRouteTable`: To associate the custom route table to the subnet.
- `ec2:ModifyVpcAttribute`: To modify the VPC's attribute to enable DNS hostnames, so that each instance launched into this VPC receives a DNS hostname.

None of the API actions in this policy support resource-level permissions, so you cannot control which specific resources users can use.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
        "ec2:CreateVpc", "ec2:CreateSubnet", "ec2:DescribeAvailabilityZones",

       "ec2:CreateRouteTable", "ec2:CreateRoute", "ec2:CreateInternetGateway",

        "ec2:AttachInternetGateway", "ec2:AssociateRouteTable", "ec2:ModifyVp
cAttribute"
      ],
      "Resource": "*"
    }
   ]
}
```

Option 2: VPC with a public and private subnet

The second VPC wizard configuration option creates a VPC with a public and private subnet, and launches a NAT instance. The following policy has the same actions as the previous example (option 1), plus actions that allow users to run and configure the NAT instance:

- `ec2:DescribeKeyPairs`: To display a list of existing key pairs in step 2 of the VPC wizard. Without this action, the wizard page cannot load.
- `ec2:DescribeImages`: To locate an AMI that's been configured to run as a NAT instance.
- `ec2:RunInstances`: To launch the NAT instance.
- `ec2:AllocateAddress` and `ec2:AssociateAddress`: To allocate an Elastic IP address to your account, and then associate it with the NAT instance.
- `ec2:ModifyInstanceAttribute`: To disable source/destination checking for the NAT instance.
- `ec2:DescribeInstances`: To check the status of the instance until it's in the running state.
- `ec2:DescribeRouteTables`, `ec2:DescribeVpnGateways`, and `ec2:DescribeVpcs`: To gather information about the routes that must be added to the main route table.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
        "ec2:CreateVpc", "ec2:CreateSubnet", "ec2:DescribeAvailabilityZones",

       "ec2:CreateRouteTable", "ec2:CreateRoute", "ec2:CreateInternetGateway",

        "ec2:AttachInternetGateway", "ec2:AssociateRouteTable", "ec2:ModifyVp
cAttribute",
        "ec2:DescribeKeyPairs", "ec2:DescribeImages", "ec2:RunInstances",
"ec2:AllocateAddress", "ec2:AssociateAddress",
        "ec2:DescribeInstances", "ec2:ModifyInstanceAttribute", "ec2:De
scribeRouteTables",
        "ec2:DescribeVpnGateways", "ec2:DescribeVpcs"
   ],
      "Resource": "*"
    }
   ]
}
```

You can use resource-level permissions on the `ec2:RunInstances` action to control users' ability to launch instances. For example, you can specify the ID of a NAT-enabled AMI so that users can only launch instances from this AMI. To find out which AMI the wizard uses to launch a NAT instance, log in to the Amazon VPC console as a user with full permissions, then carry out the second option of the VPC wizard. Switch to the Amazon EC2 console, select the **Instances** page, select the NAT instance, and note the AMI ID that was used to launch it.

The following policy allows users to launch instances using only `ami-1a2b3c4d`. If users try to launch an instance using any other AMI, the launch fails.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
        "ec2:CreateVpc", "ec2:CreateSubnet", "ec2:DescribeAvailabilityZones",

       "ec2:CreateRouteTable", "ec2:CreateRoute", "ec2:CreateInternetGateway",

        "ec2:AttachInternetGateway", "ec2:AssociateRouteTable", "ec2:ModifyVp
cAttribute",
        "ec2:DescribeKeyPairs", "ec2:DescribeImages", "ec2:AllocateAddress",
"ec2:AssociateAddress",
        "ec2:DescribeInstances", "ec2:ModifyInstanceAttribute", "ec2:De
scribeRouteTables",
        "ec2:DescribeVpnGateways", "ec2:DescribeVpcs"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:region::image/ami-1a2b3c4d",
        "arn:aws:ec2:region:account:instance/*",
        "arn:aws:ec2:region:account:subnet/*",
        "arn:aws:ec2:region:account:network-interface/*",
        "arn:aws:ec2:region:account:volume/*",
        "arn:aws:ec2:region:account:key-pair/*",
        "arn:aws:ec2:region:account:security-group/*"
      ]
    }
   ]
}
```

Option 3: VPC with public and private subnets and hardware VPN access

The third VPC wizard configuration option creates a VPC with a public and private subnet, and creates a VPN connection between your VPC and your own network. In your IAM policy, you must grant users permission to use the same actions as option 1. This allows them to create a VPC and two subnets, and to configure the routing for the public subnet. To create a VPN connection, users must also have permission to use the following actions:

- `ec2:CreateCustomerGateway`: To create a customer gateway.
- `ec2:CreateVpnGateway` and `ec2:AttachVpnGateway`: To create a virtual private gateway, and attach it to the VPC.
- `ec2:EnableVgwRoutePropagation`: To enable route propagation so that routes are automatically propagated to your route table.
- `ec2:CreateVpnConnection`: To create a VPN connection.
- `ec2:DescribeVpnConnections`, `ec2:DescribeVpnGateways`, and `ec2:DescribeCustomerGateways`: To display the options on the second configuration page of the wizard.
- `ec2:DescribeVpcs` and `ec2:DescribeRouteTables`: To gather information about the routes that must be added to the main route table.

None of the API actions in this policy support resource-level permissions, so you cannot control which specific resources users can use.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:CreateVpc", "ec2:CreateSubnet", "ec2:DescribeAvailabilityZones",
      "ec2:CreateRouteTable", "ec2:CreateRoute", "ec2:CreateInternetGateway",

     "ec2:AttachInternetGateway", "ec2:AssociateRouteTable", "ec2:ModifyVpcAt
tribute",
      "ec2:CreateCustomerGateway", "ec2:CreateVpnGateway", "ec2:AttachVpnGate
way",
      "ec2:EnableVgwRoutePropagation", "ec2:CreateVpnConnection", "ec2:De
scribeVpnGateways",
      "ec2:DescribeCustomerGateways", "ec2:DescribeVpnConnections", "ec2:De
scribeRouteTables",
      "ec2:DescribeNetworkAcls", "ec2:DescribeInternetGateways", "ec2:De
scribeVpcs"
    ],
    "Resource": "*"
  }
  ]
}
```

Option 4: VPC with a private subnet only and hardware VPN access

The fourth VPC configuration option creates a VPC with a private subnet, and creates a VPN connection between the VPC and your own network. Unlike the other three options, users do not need permission to create or attach an Internet gateway to the VPC, and they do not need permission to create a route table and associate it with the subnet. They will require the same permissions as listed in the previous example (option 3) to establish the VPN connection.

None of the API actions in this policy support resource-level permissions, so you cannot control which specific resources users can use.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:CreateVpc", "ec2:CreateSubnet", "ec2:DescribeAvailabilityZones",
     "ec2:ModifyVpcAttribute", "ec2:CreateCustomerGateway", "ec2:CreateVpnGate
way",
      "ec2:AttachVpnGateway", "ec2:EnableVgwRoutePropagation", "ec2:CreateVpn
Connection",
     "ec2:DescribeVpnGateways", "ec2:DescribeCustomerGateways", "ec2:DescribeVpn
Connections",
     "ec2:DescribeRouteTables", "ec2:DescribeNetworkAcls", "ec2:DescribeInter
netGateways", "ec2:DescribeVpcs"
    ],
    "Resource": "*"
  }
  ]
}
```

Example 2. Managing a VPC

On the **Your VPCs** page in the VPC console, you can create or delete a VPC. To view VPCs, users must have permission to use the `ec2:DescribeVPCs` action. To create a VPC using the **Create VPC** dialog box, users must have permission to use the `ec2:CreateVpc` action.

> **Note**
> By default, the VPC console creates a tag with a key of `Name` and a value that the user specifies. If users do not have permission to the use the `ec2:CreateTags` action, then they will see an error in the **Create VPC** dialog box when they try to create a VPC. However, the VPC may have been successfully created.

When you set up a VPC, you typically create a number of dependent objects, such as subnets and an Internet gateway. You cannot delete a VPC until you've disassociated and deleted these dependent objects. When you delete a VPC using the console, it performs these actions for you (except terminating your instances; you have to do this yourself).

The following example allows users to view and create VPCs on the **Your VPCs** page, and to delete VPCs that have been created with the first option in the VPC wizard - a VPC with a single public subnet. This VPC has one subnet that's associated with a custom route table, and an Internet gateway that's attached to it. To delete the VPC and its components using the console, you must grant users permission to use a number of `ec2:Describe*` actions, so that the console can check if there are any other resources that are dependent on this VPC. You must also grant users permission to disassociate the route table from the subnet, detach the Internet gateway from the VPC, and permission to delete both these resources.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
       "ec2:DescribeVpcs", "ec2:DescribeRouteTables", "ec2:DescribeVpnGateways",
 "ec2:DescribeInternetGateways",
        "ec2:DescribeSubnets", "ec2:DescribeDhcpOptions", "ec2:DescribeIn
stances", "ec2:DescribeVpcAttribute",
        "ec2:DescribeNetworkAcls",  "ec2:DescribeNetworkInterfaces", "ec2:De
scribeAddresses",
        "ec2:DescribeVpcPeeringConnections", "ec2:DescribeSecurityGroups",
        "ec2:CreateVpc", "ec2:DeleteVpc", "ec2:DetachInternetGateway",
"ec2:DeleteInternetGateway",
       "ec2:DisassociateRouteTable", "ec2:DeleteSubnet", "ec2:DeleteRouteTable"

      ],
      "Resource": "*"
    }
   ]
}
```

You can't apply resource-level permissions to any of the `ec2:Describe*` API actions, but you can apply resource-level permissions to some of the `ec2:Delete*` actions to control which resources users can delete.

For example, the following policy allows users to delete only route tables and Internet gateways that have the tag `Purpose=Test`. Users cannot delete individual route tables or Internet gateways that do not have this tag, and similarly, users cannot use the VPC console to delete a VPC that's associated with a different route table or Internet gateway.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
       "ec2:DescribeVpcs", "ec2:DescribeRouteTables", "ec2:DescribeVpnGateways",
 "ec2:DescribeInternetGateways",
        "ec2:DescribeSubnets", "ec2:DescribeDhcpOptions", "ec2:DescribeIn
stances", "ec2:DescribeVpcAttribute",
        "ec2:DescribeNetworkAcls",  "ec2:DescribeNetworkInterfaces", "ec2:De
scribeAddresses",
        "ec2:DescribeVpcPeeringConnections", "ec2:DescribeSecurityGroups",
        "ec2:CreateVpc", "ec2:DeleteVpc", "ec2:DetachInternetGateway",
        "ec2:DisassociateRouteTable", "ec2:DeleteSubnet"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action":  "ec2:DeleteInternetGateway",
      "Resource": "arn:aws:ec2:region:account:internet-gateway/*",
   "Condition": {
         "StringEquals": {
            "ec2:ResourceTag/Purpose": "Test"
         }
       }
      },
      {
      "Effect": "Allow",
      "Action": "ec2:DeleteRouteTable",
      "Resource": "arn:aws:ec2:region:account:route-table/*",
   "Condition": {
         "StringEquals": {
            "ec2:ResourceTag/Purpose": "Test"
         }
      }
    }
   ]
}
```

Example 3. Managing security groups

To view security groups on the **Security Groups** page in the Amazon VPC console, users must have permission to use the `ec2:DescribeSecurityGroups` action. To use the **Create Security Group** dialog box to create a security group, users must have permission to use the `ec2:DescribeVpcs` and `ec2:CreateSecurityGroup` actions. If users do not have permission to use the `ec2:DescribeSecurityGroups` action, they can still create a security group using the dialog box, but they may encounter an error that indicates that the group was not created.

In the **Create Security Group** dialog box, users must add the security group name and description, but they will not be able to enter a value for the **Name tag** field unless they've been granted permission to use the `ec2:CreateTags` action. However, they do not need this action to successfully create a security group.

The following policy allows users to view and create security groups, and add and remove inbound and outbound rules to any security group that's associated with `vpc-1a2b3c4d`.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeSecurityGroups", "ec2:DescribeVpcs", "ec2:CreateSecurity
Group"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteSecurityGroup", "ec2:AuthorizeSecurityGroupIngress",
"ec2:AuthorizeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress", "ec2:RevokeSecurityGroupEgress"
      ],
      "Resource": "arn:aws:ec2:*:*:security-group/*",
      "Condition":{
         "ArnEquals": {
            "ec2:Vpc": "arn:aws:ec2:*:*:vpc/vpc-1a2b3c4d"
         }
      }
    }
   ]
}
```

Example 4. Creating a VPC peering connection

To view VPC peering connections in the Amazon VPC console, users must have permission to use the `ec2:DescribePeeringConnections` action. To use the **Create VPC Peering Connection** dialog box, users must have permission to use the `ec2:DescribeVpcs` action. This allows them to view and select a VPC; without this action, the dialog box cannot load. You can apply resource-level permissions to all the `ec2:*PeeringConnection` actions, except `ec2:DescribeVpcPeeringConnections`.

The following policy allows users to view VPC peering connections, and to use the **Create VPC Peering Connection** dialog box to create a VPC peering connection using a specific requester VPC (`vpc-1a2b3c4d`) only. If users try to create a VPC peering connection with a different requester VPC, the request fails.

```
{
  "Version": "2012-10-17",
  "Statement":[{
   "Effect":"Allow",
   "Action": [
    "ec2:DescribeVpcPeeringConnections", "ec2:DescribeVpcs"
   ],
   "Resource": "*"
  },
  {
   "Effect":"Allow",
   "Action": "ec2:CreateVpcPeeringConnection",
   "Resource": [
    "arn:aws:ec2:*:*:vpc/vpc-1a2b3c4d",
    "arn:aws:ec2:*:*:vpc-peering-connection/*"
   ]
  }
  ]
}
```

For more examples of writing IAM policies for working with VPC peering connections, see 7. Creating and managing VPC peering connections (p. 94).

VPC Flow Logs

VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data is stored using Amazon CloudWatch Logs. After you've created a flow log, you can view and retrieve its data in Amazon CloudWatch Logs.

Flow logs can help you with a number of tasks; for example, to troubleshoot why specific traffic is not reaching an instance, which in turn can help you diagnose overly restrictive security group rules. You can also use flow logs as a security tool to monitor the traffic that is reaching your instance.

There is no additional charge for using flow logs; however, standard CloudWatch Logs charges apply. For more information, see Amazon CloudWatch Pricing.

Topics

- Flow Logs Basics (p. 107)
- Flow Log Limitations (p. 107)
- Flow Log Records (p. 108)
- IAM Roles for Flow Logs (p. 109)
- Working With Flow Logs (p. 110)

- Troubleshooting (p. 112)
- API and CLI Overview (p. 113)
- Examples: Flow Log Records (p. 114)
- Example: Creating a CloudWatch Metric Filter and Alarm for a Flow Log (p. 115)

Flow Logs Basics

You can create a flow log for a VPC, a subnet, or a network interface. If you create a flow log for a subnet or VPC, each network interface in the VPC or subnet is monitored. Flow log data is published to a log group in CloudWatch Logs, and each network interface has a unique log stream. Log streams contain *flow log records*, which are log events consisting of fields that describe the traffic for that network interface. For more information, see Flow Log Records (p. 108).

To create a flow log, you specify the resource for which you want to create the flow log, the type of traffic to capture (accepted traffic, rejected traffic, or all traffic), the name of a log group in CloudWatch Logs to which the flow log will be published, and the ARN of an IAM role that has sufficient permission to publish the flow log to the CloudWatch Logs log group. If you specify the name of a log group that does not exist, we'll attempt to create the log group for you. After you've created a flow log, it can take several minutes to begin collecting data and publishing to CloudWatch Logs. Flow logs do not capture real-time log streams for your network interfaces.

You can create multiple flow logs that publish data to the same log group in CloudWatch Logs. If the same network interface is present in one or more flow logs in the same log group, it has one combined log stream. If you've specified that one flow log should capture rejected traffic, and the other flow log should capture accepted traffic, then the combined log stream captures all traffic.

If you launch more instances into your subnet after you've created a flow log for your subnet or VPC, then a new log stream is created for each new network interface as soon as any network traffic is recorded for that network interface.

You can create flow logs for network interfaces that are created by other AWS services; for example, Elastic Load Balancing, Amazon RDS, Amazon ElastiCache, Amazon Redshift, and Amazon WorkSpaces. However, you cannot use these services' consoles or APIs to create the flow logs; you must use the Amazon EC2 console or the Amazon EC2 API. Similarly, you cannot use the CloudWatch Logs console or API to create log streams for your network interfaces.

If you no longer require a flow log, you can delete it. Deleting a flow log disables the flow log service for the resource, and no new flow log records or log streams are created. It does not delete any existing flow log records or log streams for a network interface. To delete an existing log stream, you can use the CloudWatch Logs console. After you've deleted a flow log, it can take several minutes to stop collecting data.

Flow Log Limitations

To use flow logs, you need to be aware of the following limitations:

- You cannot enable flow logs for network interfaces that are in the EC2-Classic platform. This includes EC2-Classic instances that have been linked to a VPC through ClassicLink.
- You cannot enable flow logs for VPCs that are peered with your VPC unless the peer VPC is in your account.
- You cannot tag a flow log.
- After you've created a flow log, you cannot change its configuration; for example, you can't associate a different IAM role with the flow log. Instead, you can delete the flow log and create a new one with the required configuration.

- None of the flow log API actions (`ec2:*FlowLogs`) support resource-level permissions. If you want to create an IAM policy to control the use of the flow log API actions, you must grant users permission to use all resources for the action by using the * wildcard for the resource element in your statement. For more information, see Controlling Access to Amazon VPC Resources (p. 87).

Flow logs do not capture all types of IP traffic. The following types of traffic are not logged:

- Traffic generated by instances when they contact the Amazon DNS server. If you use your own DNS server, then all traffic to that DNS server is logged.
- Traffic generated by a Windows instance for Amazon Windows license activation.
- Traffic to and from `169.254.169.254` for instance metadata.
- DHCP traffic.

Flow Log Records

A flow log record represents a network flow in your flow log. Each record captures the network flow for a specific 5-tuple, for a specific capture window. A 5-tuple is a set of 5 different values that specify the source, destination, and protocol for an Internet protocol (IP) flow. The capture window is a duration of time during which the flow logs service aggregates data before publishing flow log records. The capture window is approximately 10 minutes, but can take up to 15 minutes. A flow log record is a space-separated string that has the following format:

version account-id interface-id srcaddr dstaddr srcport dstport protocol packets bytes start end action log-status

Field	Description
version	The VPC flow logs version.
account-id	The AWS account ID for the flow log.
interface-id	The ID of the network interface for which the log stream applies.
srcaddr	The source IP address. The IP address of the network interface is always its private IP address.
dstaddr	The destination IP address. The IP address of the network interface is always its private IP address.
srcport	The source port of the traffic.
dstport	The destination port of the traffic.
protocol	The IANA protocol number of the traffic. For more information, go to Assigned Internet Protocol Numbers.
packets	The number of packets transferred during the capture window.
bytes	The number of bytes transferred during the capture window.
start	The time, in Unix seconds, of the start of the capture window.
end	The time, in Unix seconds, of the end of the capture window.

Field	Description
action	The action associated with the traffic: • ACCEPT: The recorded traffic was permitted by the security groups or network ACLs. • REJECT: The recorded traffic was not permitted by the security groups or network ACLs.
log-status	The logging status of the flow log: • OK: Data is logging normally to CloudWatch Logs. • NODATA: There was no network traffic to or from the network interface during the capture window. • SKIPDATA: Some flow log records were skipped during the capture window. This may be because of an internal capacity constraint, or an internal error.

If a field is not applicable for a specific record, the record displays a '-' symbol for that entry.

For examples of flow log records, see Examples: Flow Log Records (p. 114)

You can work with flow log records as you would with any other log events collected by CloudWatch Logs. For more information about monitoring log data and metric filters, see Monitoring Log Data and Filter and Pattern Syntax in the *Amazon CloudWatch Developer Guide*. For an example of setting up a metric filter and alarm for a flow log, see Example: Creating a CloudWatch Metric Filter and Alarm for a Flow Log (p. 115).

IAM Roles for Flow Logs

The IAM role that's associated with your flow log must have sufficient permissions to publish flow logs to the specified log group in CloudWatch Logs. The IAM policy that's attached to your IAM role must include at least the following permissions:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

You must also ensure that your role has a trust relationship that allows the flow logs service to assume the role (in the IAM console, choose your role, and then choose **Edit Trust Relationship** to view the trust relationship):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "vpc-flow-logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Alternatively, you can follow the procedures below to create a new role for use with flow logs.

Creating a Flow Logs Role

To create an IAM role for flow logs

1. Open the IAM console at https://console.amazonaws.cn/iam/.
2. In the navigation pane, choose **Roles**, and then choose **Create New Role**.
3. Enter a name for your role; for example, `Flow-Logs-Role`, and then choose **Next**.
4. On the **Select Role Type** page, next to **Amazon EC2**, choose **Select**.
5. On the **Attach Policy** page, choose **Next Step**.
6. On the **Review** page, take note of the ARN for your role. You will need this ARN when you create your flow log. When you are ready, choose **Create Role**.
7. Select the name of your role, expand the **Inline Policies** section, and then choose **click here**.
8. Choose **Custom Policy**, and then choose **Select**.
9. In the section IAM Roles for Flow Logs (p. 109) above, copy the first policy and paste it in the **Policy Document** window. Enter a name for your policy in the **Policy Name** field, and then choose **Apply Policy**.
10. In the section IAM Roles for Flow Logs (p. 109) above, copy the second policy (the trust relationship), and then choose **Edit Trust Relationship**. Delete the existing policy document, and paste in the new one. When you are done, choose **Update Trust Policy**.

Working With Flow Logs

You can work with flow logs using the Amazon EC2, Amazon VPC, and CloudWatch consoles.

Topics

- Creating a Flow Log (p. 111)
- Viewing Flow Logs (p. 111)
- Deleting a Flow Log (p. 112)

Creating a Flow Log

You can create a flow log from the VPC page and the Subnet page in the Amazon VPC console, or from the Network Interfaces page in the Amazon EC2 console.

To create a flow log for a network interface

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, choose **Network Interfaces**.
3. Select a network interface, choose the **Flow Logs** tab, and then **Create Flow Log**.
4. In the dialog box, complete following information. When you are done, choose **Create Flow Log**:

   - **Filter**: Select whether the flow log should capture rejected traffic, accepted traffic, or all traffic.
   - **Role**: Specify the name of an IAM role that has permission to publish logs to CloudWatch Logs.
   - **Destination Log Group**: Enter the name of a log group in CloudWatch Logs to which the flow logs will be published. You can use an existing log group, or you can enter a name for a new log group, which we'll create for you.

To create a flow log for a VPC or a subnet

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Your VPCs**, or choose **Subnets**.
3. Select your VPC or subnet, choose the **Flow Logs** tab, and then **Create Flow Log**.

   Note
   To create flow logs for multiple VPCs, choose the VPCs, and then select **Create Flow Log** from the **Actions** menu. To create flow logs for multiple subnets, choose the subnets, and then select **Create Flow Log** from the **Subnet Actions** menu.
4. In the dialog box, complete following information. When you are done, choose **Create Flow Log**:

   - **Filter**: Select whether the flow log should capture rejected traffic, accepted traffic, or all traffic.
   - **Role**: Specify the name of an IAM role that has permission to publish logs to CloudWatch Logs.
   - **Destination Log Group**: Enter the name of a log group in CloudWatch Logs to which the flow logs will be published. You can use an existing log group, or you can enter a name for a new log group, which we'll create for you.

Viewing Flow Logs

You can view information about your flow logs in the Amazon EC2 and Amazon VPC consoles by viewing the **Flow Logs** tab for a specific resource. When you select the resource, all the flow logs for that resource are listed. The information displayed includes the ID of the flow log, the flow log configuration, and information about the status of the flow log.

To view information about your flow logs for your network interfaces

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, choose **Network Interfaces**.
3. Select a network interface, and choose the **Flow Logs** tab. Information about the flow logs is displayed on the tab.

To view information about your flow logs for your VPCs or subnets

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Your VPCs**, or choose **Subnets**.
3. Select your VPC or subnet, and then choose the **Flow Logs** tab. Information about the flow logs is displayed on the tab.

You can view your flow log records using the CloudWatch Logs console. It may take a few minutes after you've created your flow log for it to be visible in the console.

To view your flow log records for a flow log

1. Open the CloudWatch console at https://console.amazonaws.cn/cloudwatch/.
2. In the navigation pane, choose **Logs**.
3. Choose the name of the log group that contains your flow log.
4. A list of log streams for each network interface is displayed. Choose the name of the log stream that contains the ID of the network interface for which you want to view the flow log records. For more information about flow log records, see Flow Log Records (p. 108).

Deleting a Flow Log

You can delete a flow log using the Amazon EC2 and Amazon VPC consoles.

> **Note**
> These procedures disable the flow log service for a resource. To delete the log streams for your network interfaces, use the CloudWatch Logs console.

To delete a flow log for a network interface

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, choose **Network Interfaces**, and then select the network interface.
3. Choose the **Flow Logs** tab, and then choose the delete button (a cross) for the flow log to delete.
4. In the confirmation dialog box, choose **Yes, Delete**.

To delete a flow log for a VPC or subnet

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Your VPCs**, or choose your **Subnets**, and then select the resource.
3. Choose the **Flow Logs** tab, and then choose the delete button (a cross) for the flow log to delete.
4. In the confirmation dialog box, choose **Yes, Delete**.

Troubleshooting

Incomplete Flow Log Records

If your flow log records are incomplete, or are no longer being published, there may be a problem delivering the flow logs to the CloudWatch Logs log group. In either the Amazon EC2 console or the Amazon VPC console, go to the **Flow Logs** tab for the relevant resource. For more information, see Viewing Flow Logs (p. 111). The flow logs table displays any errors in the **Status** column. Alternatively, use the describe-flow-logs command, and check the value that's returned in the `DeliverLogsErrorMessage` field. One of the following errors may be displayed:

- `Rate limited`: CloudWatch logs throttling has been applied. This occurs when the number of flow log records for a network interface is higher than the maximum number of records that can be published within a specific timeframe.
- `Access error`: The IAM role for your flow log does not have sufficient permissions to publish flow log records to the CloudWatch log group. For more information, see IAM Roles for Flow Logs (p. 109).
- `Unknown error`: An internal error has occurred in the flow logs service.

Flow Log is Active, But No Flow Log Records or Log Group

You've created a flow log, and the Amazon VPC or Amazon EC2 console displays the flow log as `Active`. However, you cannot see any log streams in CloudWatch Logs, or your CloudWatch Logs log group has not been created. The cause may be one of the following:

- The flow log is still in the process of being created. In some cases, it can take tens of minutes after you've created the flow log for the log group to be created, and for data to be displayed.
- There has been no traffic recorded for your network interfaces yet. The log group in CloudWatch Logs is only created when traffic is recorded.

API and CLI Overview

You can perform the tasks described on this page using the command line or API. For more information about the command line interfaces and a list of available API actions, see Accessing Amazon VPC (p. 6).

Create a flow log

- create-flow-logs (AWS CLI)
- New-EC2FlowLogs (AWS Tools for Windows PowerShell)
- CreateFlowLogs (Amazon EC2 Query API)

Describe your flow logs

- describe-flow-logs (AWS CLI)
- Get-EC2FlowLogs (AWS Tools for Windows PowerShell)
- DescribeFlowLogs (Amazon EC2 Query API)

View your flow log records (log events)

- get-log-events (AWS CLI)
- Get-CWLLogEvents (AWS Tools for Windows PowerShell)
- GetLogEvents (CloudWatch API)

Delete a flow log

- delete-flow-logs (AWS CLI)
- Remove-EC2FlowLogs (AWS Tools for Windows PowerShell)
- DeleteFlowLogs (Amazon EC2 Query API)

Examples: Flow Log Records

Flow Log Records for Accepted and Rejected Traffic

The following is an example of a flow log record in which SSH traffic (destination port 22, TCP protocol) to network interface `eni-abc123de` in account `123456789010` was allowed.

```
2 123456789010 eni-abc123de 172.168.1.12 172.168.1.11 20641 22 6 20 4249
1418530010 1418530070 ACCEPT OK
```

The following is an example of a flow log record in which RDP traffic (destination port 3389, TCP protocol) to network interface `eni-abc123de` in account `123456789010` was rejected.

```
2 123456789010 eni-abc123de 172.168.1.12 172.168.1.11 49761 3389 6 20 4249
1418530010 1418530070 REJECT OK
```

Flow Log Records for No Data and Skipped Records

The following is an example of a flow log record in which no data was recorded during the capture window.

```
2 123456789010 eni-1a2b3c4d - - - - - - - 1431280876 1431280934 - NODATA
```

The following is an example of a flow log record in which records were skipped during the capture window.

```
2 123456789010 eni-4b118871 - - - - - - - 1431280876 1431280934 - SKIPDATA
```

Security Group and Network ACL Rules

If you're using flow logs to diagnose overly restrictive or permissive security group rules or network ACL rules, then be aware of the statefulness of these resources. Security groups are stateful — this means that responses to allowed traffic are also allowed, even if the rules in your security group do not permit it. Conversely, network ACLs are stateless, therefore responses to allowed traffic are subject to network ACL rules.

For example, you use the `ping` command from your home computer (IP address is `55.123.456.78`) to your instance (the network interface's private IP address is `172.11.22.333`). Your security group's inbound rules allow ICMP traffic and the outbound rules do not allow ICMP traffic; however, because security groups are stateful, the response ping from your instance is allowed. Your network ACL permits inbound ICMP traffic but does not permit outbound ICMP traffic. Because network ACLs are stateless, the response ping is dropped and will not reach your home computer. In a flow log, this is displayed as 3 flow log entries: there are 2 `ACCEPT` entries for the originating ping and the response ping that the security group permitted, and one `REJECT` entry for response ping that the network ACL denied:

```
2 123456789010 eni-1235b8ca 55.123.456.78 172.11.22.333 0 0 1 8 672 1432917027
 1432917142 ACCEPT OK
```

```
2 123456789010 eni-1235b8ca 172.11.22.333 55.123.456.78 0 0 1 4 336 1432917027
 1432917082 ACCEPT OK
```

```
2 123456789010 eni-1235b8ca 172.11.22.333 55.123.456.78 0 0 1 4 336 1432917094
 1432917142 REJECT OK
```

Example: Creating a CloudWatch Metric Filter and Alarm for a Flow Log

In this example, you have a flow log for `eni-1a2b3c4d`. You want to create an alarm that alerts you if there have been 10 or more rejected attempts to connect to your instance over TCP port 22 (SSH) within a 1 hour time period. First, you must create a metric filter that matches the pattern of the traffic for which you want to create the alarm. Then, you can create an alarm for the metric filter.

To create a metric filter for rejected SSH traffic and create an alarm for the filter

1. Open the CloudWatch console at https://console.amazonaws.cn/cloudwatch/.
2. In the navigation pane, choose **Logs**, select the flow log group for your flow log, and then choose **Create Metric Filter**.
3. In the **Filter Pattern** field, enter the following:

```
[version, account, eni, source, destination, srcip, destip="22", protocol="6",
 packets, bytes, windowstart, windowend, action="REJECT", flowlogstatus]
```

4. In the **Select Log Data to Test** list, select the log stream for your network interface. You can optionally choose **Test Pattern** to view the lines of log data that match the filter pattern. When you're ready, choose **Assign Metric**.
5. Provide a metric namespace, a metric name, and ensure that the metric value is set to **1**. When you're done, choose **Create Filter**.
6. In the navigation pane, choose **Alarms**, and then choose **Create Alarm**.
7. In the **Custom Metrics** section, choose the namespace for the metric filter that you created.

   Note
   It can take a few minutes for a new metric to display in the console.
8. Select the metric name that you created, and then choose **Next**.
9. Enter a name and description for the alarm. In the **is** fields, choose **>=** and enter **10**. In the **for** field, leave the default **1** for the consecutive periods.
10. Choose **1 Hour** from the **Period** list, and **Sum** from the **Statistic** list. The `Sum` statistic ensures that you are capturing the total number of data points for the specified time period.
11. In the **Actions** section, you can choose to send a notification to an existing list, or you can create a new list and enter the email addresses that should receive a notification when the alarm is triggered. When you are done, choose **Create Alarm**.

Networking in Your VPC

You can use the following components to configure networking in your VPC:

- IP Addresses (p. 116)
- Network Interfaces (p. 122)
- Route Tables (p. 123)
- Internet Gateways (p. 133)
- NAT Instances (p. 138)
- DHCP Options Sets (p. 145)
- DNS (p. 149)
- VPC Peering (p. 152)
- VPC Endpoints (p. 160)

IP Addressing in Your VPC

This topic describes the IP addresses available to your Amazon EC2 instances in your VPC.

Topics

- Public and Private IP Addresses (p. 116)
- Modifying Your Subnet's Public IP Addressing Behavior (p. 117)
- Assigning a Public IP Address During Launch (p. 118)
- Elastic IP Addresses (p. 119)

Public and Private IP Addresses

We provide your instances in a VPC with IP addresses. Private IP addresses are not reachable over the Internet, and can be used for communication between the instances in your VPC. Public IP addresses are reachable over the Internet, and can be used for communication between your instances and the Internet, or with other AWS services that have public endpoints.

> **Note**
> To ensure that your instances can communicate with the Internet, you must also attach an Internet gateway to your VPC. For more information, see Internet Gateways (p. 133).

Private IP addresses

When you launch an instance into a VPC, a primary private IP address from the address range of the subnet is assigned to the default network interface (eth0) of the instance. If you don't specify a primary private IP address, we select an available IP address in the subnet range for you. For more information about network interfaces, see Elastic Network Interfaces in the *Amazon EC2 User Guide*.

You can assign additional private IP addresses, known as secondary private IP addresses, to instances that are running in a VPC. Unlike a primary private IP address, you can reassign a secondary private IP address from one network interface to another. For more information about primary and secondary IP addresses, see Multiple IP Addresses in the *Amazon EC2 User Guide*.

> **Note**
> We refer to private IP addresses as the IP addresses that are within the CIDR range of the VPC. Most VPC IP address ranges fall within the private (non-publicly routable) IP address ranges specified in RFC 1918; however, it is possible to use publicly routable CIDR blocks for your VPC. We currently do not support direct access to the Internet from publicly routable VPC CIDR blocks — if your VPC uses a publicly routable IP address range, you must set up Internet access through a virtual private gateway, a VPN connection, or AWS Direct Connect.

Public IP addresses

All subnets have an attribute that determines whether instances launched into that subnet receive a public IP address. The public IP address is assigned to the default network interface (eth0). By default, instances launched into a default subnet are assigned a public IP address. A public IP address is mapped to the primary private IP address through network address translation (NAT).

You can control whether your instance receives a public IP address by doing the following:

- Modifying the public IP addressing attribute of your subnet. For more information, see Modifying Your Subnet's Public IP Addressing Behavior (p. 117).
- Enabling or disabling the public IP addressing feature during instance launch, which overrides the subnet's public IP addressing attribute. For more information, see Assigning a Public IP Address During Launch (p. 118).

A public IP address is assigned to your instance from Amazon's pool of public IP addresses; it's not associated with your account. When a public IP address is disassociated from your instance, it's released back into the pool, and is no longer available for you to use. You cannot manually associate or disassociate a public IP address. Instead, in certain cases, we release the public IP address from your instance, or assign it a new one. For more information, see Public IP Addresses in the *Amazon EC2 User Guide for Linux Instances*.

If you require a persistent public IP address that can be assigned to and removed from instances as you require, use an Elastic IP address instead. To do this, you must allocate an Elastic IP address for use with the VPC, and then associate that Elastic IP address with a private IP address specified by the network interface attached to the instance. For more information, see Elastic IP Addresses (p. 119).

Modifying Your Subnet's Public IP Addressing Behavior

All subnets have an attribute that determines whether instances launched into that subnet are assigned a public IP address. By default, nondefault subnets have this attribute set to `false`, and default subnets have this attribute set to `true`.

You can modify the subnet's public IP addressing attribute. If you change this attribute, you can still override this setting for a specific instance during launch. For more information, see Assigning a Public IP Address During Launch (p. 118).

Note
If you use the Amazon EC2 launch wizard to launch a T2 instance type in your EC2-Classic account, and you have no VPCs, the launch wizard creates a nondefault VPC for you, with a subnet in each Availability Zone. The wizard modifies the subnets' attributes to request a public IP address for your instance automatically. For more information about T2 instance types, see T2 Instances.

To modify your subnet's public IP addressing behavior

1. Open the Amazon VPC console.
2. In the navigation pane, click **Subnets**.
3. Select your subnet, click **Subnet Actions**, and select **Modify Auto-Assign Public IP**.
4. The **Enable Auto-assign Public IP** check box, if selected, requests a public IP address for all instances launched into the selected subnet. Select or clear the check box as required, and then click **Save**.

Assigning a Public IP Address During Launch

You can control whether your instance in a default or nondefault subnet is assigned a public IP address during launch. This feature is only available if you're launching an instance with a single, new network interface with the device index of 0.

Important
You can't manually disassociate the public IP address from your instance after launch. Instead, it's automatically released in certain cases, after which you cannot reuse it. If you require a persistent public IP address that you can associate or disassociate at will, associate an Elastic IP address with the instance after launch instead. For more information, see Elastic IP Addresses (p. 119).

To access the public IP addressing feature when launching an instance

1. Open the Amazon EC2 console.
2. Click **Launch Instance**.
3. Choose an AMI and click its **Select** button, then choose an instance type and click **Next: Configure Instance Details**.
4. On the **Configure Instance Details** page, if a VPC is selected in the **Network** list, the **Auto-assign Public IP** list is displayed. Select **Enable** or **Disable** to override the default setting for the subnet.

   Important
   A public IP address can only be assigned to a single, new network interface with the device index of eth0. The **Auto-assign Public IP** list is not available if you're launching with multiple network interfaces, or if you select an existing network interface for eth0.
5. Follow the remaining steps in the wizard to launch your instance.
6. On the **Instances** screen, select your instance. On the **Description** tab, in the **Public IP** field, you can view your instance's public IP address. Alternatively, in the navigation pane, click **Network Interfaces**, and then select the eth0 network interface for your instance. You can view the public IP address in the **Public IPs** field.

   Note
   The public IP address is displayed as a property of the network interface in the console, but it's mapped to the primary private IP address through NAT. Therefore, if you inspect the properties of your network interface on your instance, for example, through `ipconfig` on a Windows instance, or `ifconfig` on a Linux instance, the public IP address is not displayed. To determine your instance's public IP address from within the instance, you can use instance metadata. For more information, see Instance Metadata and User Data.

This feature is only available during launch. However, whether you assign a public IP address to your instance during launch or not, you can associate an Elastic IP address with your instance after it's launched. For more information, see Elastic IP Addresses (p. 119).

Elastic IP Addresses

An *Elastic IP address* is a static, public IP address designed for dynamic cloud computing. You can associate an Elastic IP address with any instance or network interface for your VPC. With an Elastic IP address, you can mask the failure of an instance by rapidly remapping the address to another instance in your VPC. Note that the advantage of associating the Elastic IP address with the network interface instead of directly with the instance is that you can move all the attributes of the network interface from one instance to another in a single step.

Topics

- Elastic IP Address Basics (p. 119)
- Working with Elastic IP Addresses (p. 120)
- API and Command Overview (p. 121)

Elastic IP Address Basics

The following are the basic things that you need to know about Elastic IP addresses:

- You first allocate an Elastic IP address for use in a VPC, and then associate it with an instance in your VPC (it can be assigned to only one instance at a time).
- An Elastic IP address is a property of network interfaces. You can associate an Elastic IP address with an instance by updating the network interface attached to the instance.
- If you associate an Elastic IP address with the eth0 network interface of your instance, its current public IP address (if it had one) is released to the EC2-VPC public IP address pool. If you disassociate the Elastic IP address, the eth0 network interface is automatically assigned a new public IP address within a few minutes. This doesn't apply if you've attached a second network interface to your instance.
- There are differences between an Elastic IP address that you use in a VPC and one that you use in EC2-Classic. For more information, see Differences Between EC2-Classic and Amazon EC2-VPC in the *Amazon EC2 User Guide for Linux Instances*).
- You can move an Elastic IP address from one instance to another. The instance can be in the same VPC or another VPC, but not in EC2-Classic.
- Your Elastic IP addresses remain associated with your AWS account until you explicitly release them.
- To ensure efficient use of Elastic IP addresses, we impose a small hourly charge when they aren't associated with a running instance, or when they are associated with a stopped instance or an unattached network interface. While your instance is running, you aren't charged for one Elastic IP address associated with the instance, but you are charged for any additional Elastic IP addresses associated with the instance. For more information, see Amazon EC2 Pricing.
- You're limited to 5 Elastic IP addresses; to help conserve them, you can use a NAT instance (see NAT Instances (p. 138)).
- An Elastic IP address is accessed through the Internet gateway of a VPC. If you have set up a VPN connection between your VPC and your network, the VPN traffic traverses a virtual private gateway, not an Internet gateway, and therefore cannot access the Elastic IP address.
- You can migrate an Elastic IP address that you've allocated for use in the EC2-Classic platform to the VPC platform. For more information, see Migrating an Elastic IP Address from EC2-Classic to EC2-VPC in the *Amazon EC2 User Guide*.

Working with Elastic IP Addresses

You can allocate an Elastic IP address and then associate it with an instance in a VPC.

To allocate an Elastic IP address for use in a VPC

1. Open the Amazon VPC console.
2. In the navigation pane, click **Elastic IPs**.
3. Click the **Allocate New Address** button.
4. Click **Yes, Allocate**.

   Note
   If your account supports EC2-Classic, first choose **EC2-VPC** from the **Network platform** list.

To view your Elastic IP addresses

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. To filter the displayed list, start typing part of the Elastic IP address or the ID of the instance to which it's assigned in the search box.

To associate an Elastic IP address with a running instance in a VPC

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. Select an Elastic IP address that's allocated for use with a VPC (the **Scope** column has a value of `vpc`), and then click the **Associate Address** button.
4. In the **Associate Address** dialog box, select **Instance** or **Network Interface** from the **Associate with** list, and then either the instance or network interface ID. Select the private IP address to associate the Elastic IP address with from the **Private IP address** list, and then click **Yes, Associate**.

   Note
   A network interface can have several attributes, including an Elastic IP address. You can create a network interface and attach and detach it from instances in your VPC. The advantage of making the Elastic IP address an attribute of the network interface instead of associating it directly with the instance is that you can move all the attributes of the network interface from one instance to another in a single step. For more information, see Elastic Network Interfaces.
5. (Optional) After you associate the Elastic IP address with your instance, it receives a DNS hostname if DNS hostnames are enabled. For more information, see Using DNS with Your VPC (p. 149).

To change which instance an Elastic IP address is associated with, disassociate it from the currently associated instance, and then associate it with the new instance in the VPC.

To disassociate an Elastic IP address

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. Select the Elastic IP address, and then click the **Disassociate Address** button.
4. When prompted, click **Yes, Disassociate**.

If you no longer need an Elastic IP address, we recommend that you release it (the address must not be associated with an instance). You incur charges for any Elastic IP address that's allocated for use with a VPC but not associated with an instance.

To release an Elastic IP address

1. Open the Amazon VPC console.
2. Click **Elastic IPs** in the navigation pane.
3. Select the Elastic IP address, and then click the **Release Address** button.
4. When prompted, click **Yes, Release**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Acquire an Elastic IP address

- allocate-address (AWS CLI)
- ec2-allocate-address (Amazon EC2 CLI)
- New-EC2Address (AWS Tools for Windows PowerShell)

Associate an Elastic IP address with an instance or network interface

- associate-address (AWS CLI)
- ec2-associate-address (Amazon EC2 CLI)
- Register-EC2Address (AWS Tools for Windows PowerShell)

Describe one or more Elastic IP addresses

- describe-addresses (AWS CLI)
- ec2-describe-addresses (Amazon EC2 CLI)
- Get-EC2Address (AWS Tools for Windows PowerShell)

Disassociate an Elastic IP address

- disassociate-address (AWS CLI)
- ec2-disassociate-address (Amazon EC2 CLI)
- Unregister-EC2Address (AWS Tools for Windows PowerShell)

Release an Elastic IP address

- release-address (AWS CLI)
- ec2-release-address (Amazon EC2 CLI)
- Remove-EC2Address (AWS Tools for Windows PowerShell)

Assign a public IP address during launch

- Use the `--associate-public-ip-address` or the `--no-associate-public-ip-address` option with the run-instances command. (AWS CLI)

- Use the `--associate-public-ip-address` option with the ec2-run-instances command. (Amazon EC2 CLI)
- Use the `-AssociatePublicIp` parameter with the New-EC2Instance command. (AWS Tools for Windows PowerShell)

Modify a subnet's public IP addressing behavior

- modify-subnet-attribute (AWS CLI)
- ec2-modify-subnet-attribute (Amazon EC2 CLI)
- Edit-EC2SubnetAttribute (AWS Tools for Windows PowerShell)

Using Elastic Network Interfaces with Your VPC

An elastic network interface (ENI) is a virtual network interface that can include the following attributes:

- a primary private IP address
- one or more secondary private IP addresses
- one Elastic IP address per private IP address
- one public IP address, which can be auto-assigned to the network interface for eth0 when you launch an instance, but only when you create a network interface for eth0 instead of using an existing network interface
- one or more security groups
- a MAC address
- a source/destination check flag
- a description

You can create an ENI, attach it to an instance, detach it from an instance, and attach it to another instance. An ENI's attributes follow the ENI as it is attached or detached from an instance and reattached to another instance. When you move an ENI from one instance to another, network traffic is redirected to the new instance.

Each instance in your VPC has a default elastic network interface (the primary network interface) that is assigned a private IP address from the IP address range of your VPC. You cannot detach a primary network interface from an instance. You can create and attach an additional elastic network interface to any instance in your VPC. The number of ENIs you can attach varies by instance type. For more information, see Private IP Addresses Per ENI Per Instance Type in the *Amazon EC2 User Guide for Linux Instances*.

Attaching multiple ENIs to an instance is useful when you want to:

- Create a management network.
- Use network and security appliances in your VPC.
- Create dual-homed instances with workloads/roles on distinct subnets.
- Create a low-budget, high-availability solution.

For more information about ENIs, and step-by-step instructions for working with them using the Amazon EC2 console, see Elastic Network Interfaces in the *Amazon EC2 User Guide for Linux Instances*.

Route Tables

A *route table* contains a set of rules, called *routes*, that are used to determine where network traffic is directed.

Each subnet in your VPC must be associated with a route table; the table controls the routing for the subnet. A subnet can only be associated with one route table at a time, but you can associate multiple subnets with the same route table.

Topics

- Route Table Basics (p. 123)
- Main Route Tables (p. 123)
- Custom Route Tables (p. 124)
- Route Table Association (p. 125)
- Route Tables for VPC Peering Connections (p. 126)
- Route Tables for ClassicLink (p. 127)
- Route Tables for VPC Endpoints (p. 127)
- Working with Route Tables (p. 127)
- API and Command Overview (p. 131)

Route Table Basics

The following are the basic things that you need to know about route tables:

- Your VPC has an implicit router.
- Your VPC automatically comes with a main route table that you can modify.
- You can create additional custom route tables for your VPC.
- Each subnet must be associated with a route table, which controls the routing for the subnet. If you don't explicitly associate a subnet with a particular route table, the subnet uses the main route table.
- You can replace the main route table with a custom table that you've created (so that this table is the default table each new subnet is associated with).
- Each route in a table specifies a destination CIDR and a target (for example, traffic destined for 172.16.0.0/12 is targeted for the virtual private gateway); we use the most specific route that matches the traffic to determine how to route the traffic.

Main Route Tables

When you create a VPC, it automatically has a main route table. On the **Route Tables** page in the VPC console, you can view the main route table for a VPC by looking for **Yes** in the **Main** column.

Initially, the main route table (and every route table in a VPC) contains only a single route: a local route that enables communication within the VPC.

You can't modify the local route in a route table. Whenever you launch an instance in the VPC, the local route automatically covers that instance; you don't need to add the new instance to a route table.

If you don't explicitly associate a subnet with a route table, the subnet is implicitly associated with the main route table. However, you can still explicitly associate a subnet with the main route table. You might do that if you change which table is the main route table (see Replacing the Main Route Table (p. 130)).

The console shows the number of subnets associated with each table. Only explicit associations are included in that number (see Determining Which Subnets Are Explicitly Associated with a Table (p. 128)).

When you add a gateway to a VPC (either an Internet gateway or a virtual private gateway), you must update the route table for any subnet that uses that gateway.

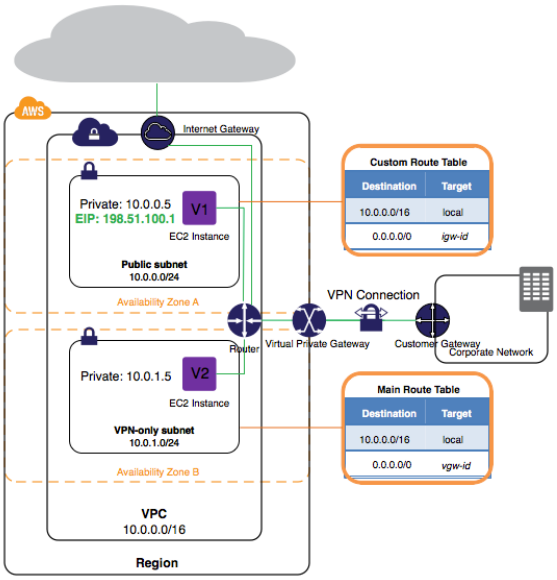
If you've attached a virtual private gateway to your VPC and enabled route propagation on your route table, routes representing your VPN connection automatically appear as propagated routes in your route table's list of routes.

Custom Route Tables

Your VPC can have route tables other than the default table. One way to protect your VPC is to leave the main route table in its original default state (with only the local route), and explicitly associate each new subnet you create with one of the custom route tables you've created. This ensures that you must explicitly control how each subnet's outbound traffic is routed.

For information about the limit on the number of route tables that you can create, see Amazon VPC Limits (p. 194).

The following diagram shows the routing for a VPC with both an Internet gateway and a virtual private gateway, plus a public subnet and a VPN-only subnet. The main route table came with the VPC, and it also has a route for the VPN-only subnet. There's a custom route table that's associated with the public subnet. The custom route table has a route for the public subnet over the Internet gateway (the destination is 0.0.0.0/0, and the target is the Internet gateway).



If you create a new subnet in this VPC, it would be automatically associated with the main route table, which routes its traffic to the virtual private gateway. If you were to set up the reverse configuration (the main route table with the route to the Internet gateway, and the custom route table with the route to the virtual private gateway), then if you create a new subnet, it would automatically have a route to the Internet gateway.

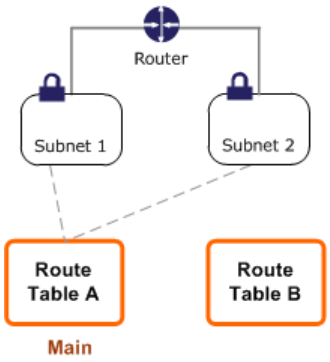
Route Table Association

The main route table is the default table that subnets use if they're not explicitly associated with another table. When you add a new subnet, it automatically uses the routes specified in the main route table. You can change which table is the main route table, and thus change the default for additional new subnets.

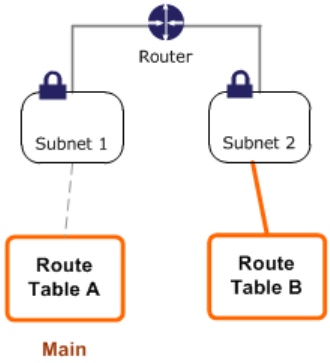
Subnets can be implicitly or explicitly associated with the main route table. Subnets typically won't have an explicit association to the main route table, although it might happen temporarily if you're replacing the main route table.

You might want to make changes to the main route table, but to avoid any disruption to your traffic, you decide to first test the route changes using a custom route table. After you're satisfied with the testing, you then replace the main route table with the new custom table.

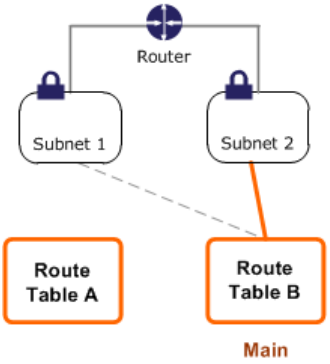
The following diagram shows a VPC with two subnets that are implicitly associated with the main route table (Route Table A), and a custom route table (Route Table B) that isn't associated with any subnets.



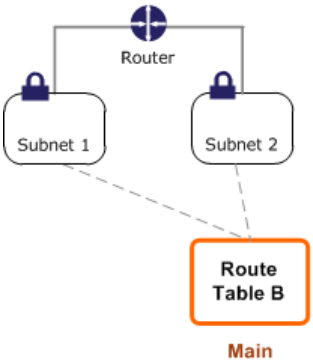
You can create an explicit association between Subnet 2 and Route Table B.



After you've tested Route Table B, you can make it the main route table. Note that Subnet 2 still has an explicit association with Route Table B, and Subnet 1 has an implicit association with Route Table B because it is the new main route table. Route Table A is no longer in use.



If you disassociate Subnet 2 from Route Table B, there's still an implicit association between Subnet 2 and Route Table B. If you no longer need Route Table A, you can delete it.



Route Tables for VPC Peering Connections

A VPC peering connection is a networking connection between two VPCs that allows you to route traffic between them using private IP addresses. Instances in either VPC can communicate with each other as if they are part of the same network.

To enable the routing of traffic between VPCs in a VPC peering connection, you must add a route to one or more of your VPC's route tables that points to the VPC peering connection to access all or part of the CIDR block of the other VPC in the peering connection. Similarly, the owner of the other VPC must add a route to their VPC's route table to route traffic back to your VPC.

For example, you have a VPC peering connection (`pcx-1a2b1a2b`) between two VPCs, with the following information:

- VPC A: `vpc-1111aaaa`, CIDR block is 10.0.0.0/16
- VPC B: `vpc-2222bbbb`, CIDR block is 172.31.0.0/16

To enable traffic between the VPCs and allow access to the entire CIDR block of either VPC, VPC A's route table is configured as follows:

Destination	Target
10.0.0.0/16	Local
172.31.0.0/16	pcx-1a2b1a2b

VPC B's route table is configured as follows:

Destination	Target
172.31.0.0/16	Local
10.0.0.0/16	pcx-1a2b1a2b

For more information about VPC peering connections, see the following topics:

- Working with VPC peering connections in the VPC console: VPC Peering (p. 152)
- Adding routes for VPC peering connections: Updating Route Tables for Your VPC Peering Connection (p. 157)
- Supported VPC peering connection scenarios and routing configurations: Amazon VPC Peering Guide

Route Tables for ClassicLink

ClassicLink is a feature that enables you to link an EC2-Classic instance to a VPC, allowing communication between the EC2-Classic instance and instances in the VPC using private IP addresses. For more information about ClassicLink, see ClassicLink (p. 193).

When you enable a VPC for ClassicLink, a route is added to all of the VPC's route tables with a destination of `10.0.0.0/8` and a target of `local`. This allows communication between instances in the VPC and any EC2-Classic instances that are then linked to the VPC. If you add another route table to a ClassicLink-enabled VPC, it automatically receives a route with a destination of `10.0.0.0/8` and a target of `local`. If you disable ClassicLink for a VPC, this route is automatically deleted in all the VPC's route tables.

If any of your VPC's route tables have existing routes for address ranges within the `10.0.0.0/8` CIDR, then you cannot enable your VPC for ClassicLink. This does not include local routes for VPCs with `10.0.0.0/16` and `10.1.0.0/16` IP address ranges.

If you've already enabled a VPC for ClassicLink, you may not be able to add any more specific routes to your route tables for the `10.0.0.0/8` IP address range.

Route Tables for VPC Endpoints

A VPC endpoint enables you to create a private connection between your VPC and another AWS service. When you create an endpoint, you specify the route tables in your VPC that are used by the endpoint. A route is automatically added to each of the route tables with a destination that specifies the prefix list ID of the service (`pl-`*`xxxxxxxx`*), and a target with the endpoint ID (`vpce-`*`xxxxxxxx`*). You cannot explicitly delete or modify the endpoint route, but you can change the route tables that are used by the endpoint.

For more information about routing for endpoints, and the implications for routes to AWS services, see Routing for Endpoints (p. 161).

Working with Route Tables

This section shows you how to work with route tables.

> **Note**
> When you use the wizard in the console to create a VPC with a gateway, the wizard automatically updates the route tables to use the gateway. If you're using the command line tools or API to set up your VPC, you must update the route tables yourself.

Topics

- Determining Which Route Table a Subnet Is Associated With (p. 128)
- Determining Which Subnets Are Explicitly Associated with a Table (p. 128)
- Creating a Custom Route Table (p. 129)
- Adding and Removing Routes from a Route Table (p. 129)
- Enabling and Disabling Route Propagation (p. 129)
- Associating a Subnet with a Route Table (p. 130)
- Changing a Subnet's Route Table (p. 130)
- Disassociating a Subnet from a Route Table (p. 130)
- Replacing the Main Route Table (p. 130)
- Deleting a Route Table (p. 131)

Determining Which Route Table a Subnet Is Associated With

You can determine which route table a subnet is associated with by looking at the subnet's details in the Amazon VPC Console.

To determine which route table a subnet is associated with

1. Open the Amazon VPC console.
2. Click **Subnets** in the navigation pane, and then select the subnet.

   The subnet details are displayed in the **Summary** tab. Click the **Route Table** tab to view the route table ID and its routes. If it's the main route table, the console doesn't indicate whether the association is implicit or explicit. To determine if the association to the main route table is explicit, see Determining Which Subnets Are Explicitly Associated with a Table (p. 128).

Determining Which Subnets Are Explicitly Associated with a Table

You can determine how many and which subnets are explicitly associated with a route table.

The main route table can have explicit and implicit associations. Custom route tables have only explicit associations.

Subnets that aren't explicitly associated with any route table have an implicit association with the main route table. You can explicitly associate a subnet with the main route table (for an example of why you might do that, see Replacing the Main Route Table (p. 130)).

To determine how many subnets are explicitly associated

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. Click **Route Tables** in the navigation pane.
3. Check the **Explicitly Associated With** column to determine the number of explicitly associated subnets.

To determine which subnets are explicitly associated

1. Select the route table of interest.

2. Click the **Subnet Associations** tab in the details pane. The subnets explicitly associated with the table are listed on the tab. Any subnets not associated with any route table (and thus implicitly associated with the main route table) are also listed.

Creating a Custom Route Table

Depending on your situation, you might need to create your own route tables.

To create a custom route table

1. Open the Amazon VPC console.
2. In the navigation pane, click **Route Tables**.
3. Click **Create Route Table**.
4. In the **Create Route Table** dialog box, you can optionally name your route table in the **Name tag** field. Doing so creates a tag with a key of `Name` and a value that you specify. Select your VPC from the **VPC** list, and then click **Yes, Create**.

Adding and Removing Routes from a Route Table

You can add, delete and modify routes in your route tables. You can only modify routes that you've added.

To modify or add a route to a route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Routes** tab, click **Edit**.
4. To modify an existing route, replace the destination CIDR block or a single IP address in the **Destination** field, and then select a target from the **Target** list. Click **Add another route** to add more routes, and then click **Save** when you're done.

To delete a route from a route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Routes** tab, click **Edit**, and then click the **Remove** button for the route you want to delete.
4. Click **Save** when you're done.

Enabling and Disabling Route Propagation

Route propagation allows a virtual private gateway to automatically propagate routes to the route tables so that you don't need to manually enter VPN routes to your route tables. You can enable or disable route propagation.

For more information about VPN routing options, see VPN Routing Options (p. 174).

To enable route propagation

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the details pane, click the **Route Propagation** tab.
4. Click **Edit**, and then select the virtual private gateway. Click **Save**.

To disable route propagation

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. On the **Route Propagation** tab, click **Edit**, and then deselect the **Propagate** check box next to the ID of the virtual private gateway. Click **Save**.

Associating a Subnet with a Route Table

To apply a route table's routes to a particular subnet, you must associate the route table with the subnet. A route table can be associated with multiple subnets; however, a subnet can only be associated with one route table at a time. Any subnet not explicitly associated with a table is implicitly associated with the main route table by default.

To associate a table with a subnet

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the details pane, on the **Subnet Associations** tab, click **Edit**.
4. Select the **Associate** check box for the subnet to associate with the route table, and then click **Save**.

Changing a Subnet's Route Table

You can change which route table a subnet is associated with. For example, when you create a subnet, it is implicitly associated with the main route table. You might want to instead associate it with a custom route table you've created.

To change a subnet's route table association

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Subnets**, and then select the subnet.
3. In the **Route Table** tab, click **Edit**.
4. Select the new route table to associate the subnet with from the **Change to** list, and then click **Save**.

Disassociating a Subnet from a Route Table

You might want to disassociate a subnet from a route table. For example, you might have a subnet that is associated with a custom route table, and you instead want it associated with the main route table. By disassociating the subnet from the custom route table, the subnet becomes implicitly associated with the main route table.

To disassociate a subnet from a route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Subnet Associations** tab, click **Edit**.
4. Deselect the **Associate** check box for the subnet, and then click **Save**.

Replacing the Main Route Table

The following procedure describes how to change which route table is the main route table in your VPC.

To replace the main route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. Click **Route Tables** in the navigation pane.
3. Locate the route table that you want to be the new main route table, right-click the table, and then select **Set as Main Table**.
4. In the **Set Main Route Table** dialog box, click **Yes, Set**.

The following procedure describes how to remove an explicit association between a subnet and the main route table. The result is an implicit association between the subnet and the main route table. The process is the same as disassociating any subnet from any route table.

To remove an explicit association with the main route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**, and then select the route table.
3. In the **Subnet Associations** tab, click **Edit**.
4. Deselect the **Associate** check box for the subnet, and then click **Save**.

Deleting a Route Table

You can delete a route table only if there are no subnets associated with it. You can't delete the main route table.

To delete a route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**.
3. Select the route table, and then click the **Delete** button.
4. In the **Delete Route Table** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Create a custom route table

- create-route-table (AWS CLI)
- ec2-create-route-table (Amazon EC2 CLI)
- New-EC2RouteTable (AWS Tools for Windows PowerShell)

Add a route to a route table

- create-route (AWS CLI)
- ec2-create-route (Amazon EC2 CLI)
- New-EC2Route (AWS Tools for Windows PowerShell)

Associate a subnet with a route table

- associate-route-table (AWS CLI)
- ec2-associate-route-table (Amazon EC2 CLI)
- Register-EC2RouteTable (AWS Tools for Windows PowerShell)

Describe one or more route tables

- describe-route-tables (AWS CLI)
- ec2-describe-route-tables (Amazon EC2 CLI)
- Get-EC2RouteTable (AWS Tools for Windows PowerShell)

Delete a route from a route table

- delete-route (AWS CLI)
- ec2-delete-route (Amazon EC2 CLI)
- Remove-EC2Route (AWS Tools for Windows PowerShell)

Replace an existing route in a route table

- replace-route (AWS CLI)
- ec2-replace-route (Amazon EC2 CLI)
- Set-EC2Route (AWS Tools for Windows PowerShell)

Disassociate a subnet from a route table

- disassociate-route-table (AWS CLI)
- ec2-disassociate-route-table (Amazon EC2 CLI)
- Unregister-EC2RouteTable (AWS Tools for Windows PowerShell)

Change the route table associated with a subnet

- replace-route-table-association (AWS CLI)
- ec2-replace-route-table-association (Amazon EC2 CLI)
- Set-EC2RouteTableAssociation (AWS Tools for Windows PowerShell)

Create a static route associated with a VPN connection

- create-vpn-connection-route (AWS CLI)
- ec2-create-vpn-connection-route (Amazon EC2 CLI)
- New-EC2VpnConnectionRoute (AWS Tools for Windows PowerShell)

Delete a static route associated with a VPN connection

- delete-vpn-connection-route (AWS CLI)
- ec2-delete-vpn-connection-route (Amazon EC2 CLI)
- Remove-EC2VpnConnectionRoute (AWS Tools for Windows PowerShell)

Enable a virtual private gateway (VGW) to propagate routes to the routing tables of a VPC

- enable-vgw-route-propagation (AWS CLI)
- ec2-enable-vgw-route-propagation (Amazon EC2 CLI)
- Enable-EC2VgwRoutePropagation (AWS Tools for Windows PowerShell)

Disable a VGW from propagating routes to the routing tables of a VPC

- disable-vgw-route-propagation (AWS CLI)
- ec2-disable-vgw-route-propagation (Amazon EC2 CLI)
- Disable-EC2VgwRoutePropagation (AWS Tools for Windows PowerShell)

Delete a route table

- delete-route-table (AWS CLI)
- ec2-delete-route-table (Amazon EC2 CLI)
- Remove-EC2RouteTable (AWS Tools for Windows PowerShell)

Internet Gateways

An Internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the Internet. It therefore imposes no availability risks or bandwidth constraints on your network traffic. An Internet gateway serves two purposes: to provide a target in your VPC route tables for Internet-routable traffic, and to perform network address translation (NAT) for instances that have been assigned public IP addresses.

Enabling Internet Access

To enable access to or from the Internet for instances in a VPC subnet, you must do the following:

- Attach an Internet gateway to your VPC.
- Ensure that your subnet's route table points to the Internet gateway.
- Ensure that instances in your subnet have public IP addresses or Elastic IP addresses.
- Ensure that your network access control and security group rules allow the relevant traffic to flow to and from your instance.

To use an Internet gateway, your subnet's route table must contain a route that directs Internet-bound traffic to the Internet gateway. You can scope the route to all destinations not explicitly known to the route table (`0.0.0.0/0`), or you can scope the route to a narrower range of IP addresses; for example, the public IP addresses of your company's public endpoints outside of AWS, or the Elastic IP addresses of other Amazon EC2 instances outside your VPC. If your subnet is associated with a route table that has a route to an Internet gateway, it's known as a public subnet. However, the Internet gateway route is not sufficient to provide Internet access to instances in the subnet. For more information about public and private subnets, see Your VPC with Subnets (p. 51).

To enable an instance in your public subnet to communicate with the Internet, it must have a public IP address or an Elastic IP address that's associated with a private IP address on your instance. Your instance is only aware of the private (internal) IP address space defined within the VPC and subnet. The Internet gateway logically provides the one-to-one NAT on behalf of your instance, so that when traffic leaves your VPC subnet and goes to the Internet, the reply address field is set to the public IP address or Elastic IP address of your instance, and not its private IP address. Conversely, traffic that's destined

for public IP address or Elastic IP address of your instance has its destination address translated into the instance's private IP address before the traffic is delivered to the VPC.

Internet Access for Default and Nondefault VPCs

Your default VPC comes with an Internet gateway, and instances launched into a default subnet receive a public IP address by default, unless you specify otherwise during launch, or you modify the subnet's public IP address attribute. Therefore, instances that you launch into a default subnet can automatically communicate with the Internet. For more information, see Your Default VPC and Subnets (p. 57).

Instances that you launch into a nondefault subnet may or may not be able to communicate with the Internet, depending on how you create and configure your VPC. For example, if you use the VPC wizard to create your VPC, depending on the option that you select, the VPC wizard adds an Internet gateway to your VPC and updates the route table so that your instances can communicate with the Internet. For more information about using the VPC wizard to create a subnet with an Internet gateway, see Scenario 1: VPC with a Single Public Subnet (p. 17) or Scenario 2: VPC with Public and Private Subnets (NAT) (p. 22). Instances that you launch into a nondefault subnet do not receive a public IP address by default and therefore can't communicate with the Internet, unless you specifically assign one during launch, or you modify the subnet's public IP address attribute. For more information about assigning a public IP address at launch, see Assigning a Public IP Address During Launch (p. 118). For more information about modifying your subnet's public IP addressing attribute, see Modifying Your Subnet's Public IP Addressing Behavior (p. 117).

When you add a new subnet to your VPC, you must set up the routing and security that you want for the subnet.

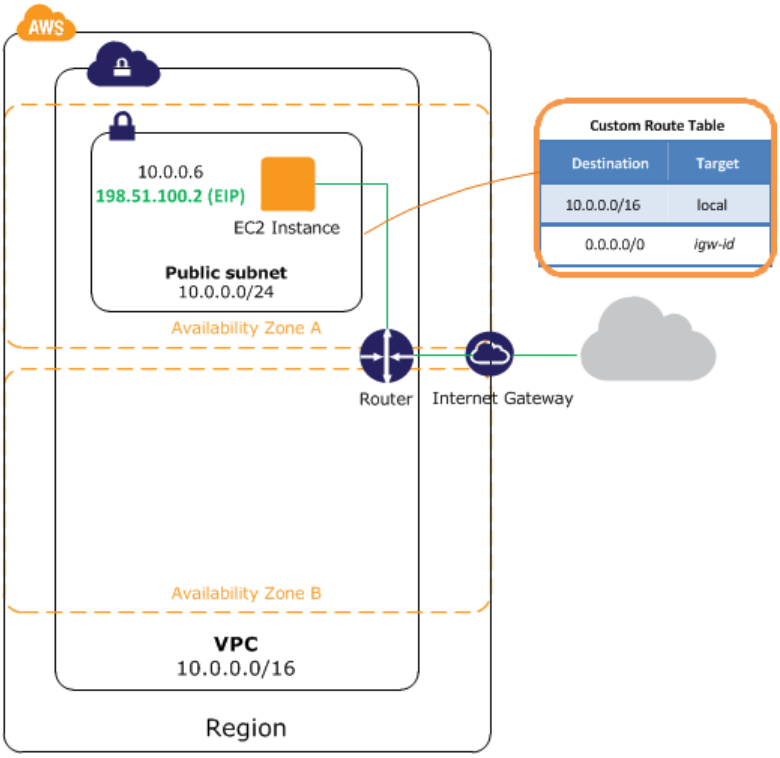
Creating a VPC with an Internet Gateway

The following sections describe how to manually create a public subnet to support Internet access.

Topics

- Creating a Subnet (p. 135)
- Attaching an Internet Gateway (p. 135)
- Creating a Custom Route Table (p. 135)
- Updating the Security Group Rules (p. 136)
- Adding Elastic IP Addresses (p. 136)
- Detaching an Internet Gateway from Your VPC (p. 137)
- Deleting an Internet Gateway (p. 137)
- API and Command Overview (p. 137)

When you are finished setting up the subnet, your VPC is configured as shown in the following diagram.



Creating a Subnet

To add a subnet to your VPC

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Subnets**, and then choose **Create Subnet**.
3. In the **Create Subnet** dialog box, select the VPC, select the Availability Zone, specify the CIDR range for the subnet, and then choose **Yes, Create**.

For more information about subnets, see Your VPC and Subnets (p. 47).

Attaching an Internet Gateway

To create an Internet gateway and attach it to your VPC

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Internet Gateways**, and then choose **Create Internet Gateway**.
3. In the **Create Internet Gateway** dialog box, you can optionally name your Internet gateway, and then choose **Yes, Create**.
4. Select the Internet gateway that you just created, and then choose **Attach to VPC**.
5. In the **Attach to VPC** dialog box, select your VPC from the list, and then choose **Yes, Attach**.

Creating a Custom Route Table

When you create a subnet, we automatically associate it with the main route table for the VPC. By default, the main route table doesn't contain a route to an Internet gateway. The following procedure creates a

custom route table with a route that sends traffic destined outside the VPC to the Internet gateway, and then associates it with your subnet.

To create a custom route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Route Tables**, and then choose **Create Route Table**.
3. In the **Create Route Table** dialog box, optionally name your route table, then select your VPC, and then choose **Yes, Create**.
4. Select the custom route table that you just created. The details pane displays tabs for working with its routes, associations, and route propagation.
5. On the **Routes** tab, choose **Edit**, specify `0.0.0.0/0` in the **Destination** box, select the Internet gateway ID in the **Target** list, and then choose **Save**.
6. On the **Subnet Associations** tab, choose **Edit**, select the **Associate** check box for the subnet, and then choose **Save**.

For more information about route tables, see Route Tables (p. 123).

Updating the Security Group Rules

Your VPC comes with a default security group. Each instance that you launch into a VPC is automatically associated with its default security group. The default settings for a default security group allow no inbound traffic from the Internet and allow all outbound traffic to the Internet. Therefore, to enable your instances to communicate with the Internet, create a new security group that allows public instances to access the Internet.

To create a new security group and associate it with your instances

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Security Groups**, and then choose **Create Security Group**.
3. In the **Create Security Group** dialog box, specify a name for the security group and a description. Select the ID of your VPC from the **VPC** list, and then choose **Yes, Create**.
4. Select the security group. The details pane displays the details for the security group, plus tabs for working with its inbound rules and outbound rules.
5. On the **Inbound Rules** tab, choose **Edit**. Choose **Add Rule**, and complete the required information. For example, select **HTTP** or **HTTPS** from the **Type** list, and enter the **Source** as `0.0.0.0/0`. Choose **Save** when you're done.
6. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
7. In the navigation pane, choose **Instances** .
8. Select the instance, choose **Actions**, then **Networking**, and then select **Change Security Groups**.
9. In the **Change Security Groups** dialog box, clear the check box for the currently selected security group, and select the new one. Choose **Assign Security Groups**.

For more information about security groups, see Security Groups for Your VPC (p. 64).

Adding Elastic IP Addresses

After you've launched an instance into the subnet, you must assign it an Elastic IP address if you want it to be reachable from the Internet.

Note
If you assigned a public IP address to your instance during launch, then your instance is reachable from the Internet, and you do not need to assign it an Elastic IP address. For more information about IP addressing for your instance, see IP Addressing in Your VPC (p. 116).

To allocate an Elastic IP address and assign it to an instance using the console

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Elastic IPs**.
3. Choose **Allocate New Address**.
4. In the **Allocate New Address** dialog box, in the **Network platform** list, select **EC2-VPC**, and then choose **Yes, Allocate**.
5. Select the Elastic IP address from the list, and then choose **Associate Address**.
6. In the **Associate Address** dialog box, select **Instance** or **Network Interface** from the **Associate with** list, and then either the instance or network interface ID. Select the private IP address to associate the Elastic IP address with from the **Private IP address** list, and then choose **Yes, Associate**.

For more information about Elastic IP addresses, see Elastic IP Addresses (p. 119).

Detaching an Internet Gateway from Your VPC

If you no longer need Internet access for instances that you launch into a nondefault VPC, you can detach an Internet gateway from a VPC. You can't detach an Internet gateway if the VPC has instances with associated Elastic IP addresses.

To detach an Internet gateway

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Elastic IPs**.
3. Select the IP address, choose **Disassociate Address**, and then choose **Yes, Disassociate**.
4. In the navigation pane, choose **Internet Gateways** .
5. Select the Internet gateway and choose **Detach from VPC**.
6. In the **Detach from VPC** dialog box, choose **Yes, Detach**.

Deleting an Internet Gateway

If you no longer need an Internet gateway, you can delete it. You can't delete an Internet gateway if it's still attached to a VPC.

To delete an Internet gateway

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Internet Gateways**.
3. Select the Internet gateway and choose **Delete**.
4. In the **Delete Internet Gateway** dialog box, choose **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available API actions, see Accessing Amazon VPC (p. 6).

Create an Internet gateway

- create-internet-gateway (AWS CLI)
- ec2-create-internet-gateway (Amazon EC2 CLI)
- New-EC2InternetGateway (AWS Tools for Windows PowerShell)

Attach an Internet gateway to a VPC

- attach-internet-gateway (AWS CLI)
- ec2-attach-internet-gateway (Amazon EC2 CLI)
- Add-EC2InternetGateway (AWS Tools for Windows PowerShell)

Describe an Internet gateway

- describe-internet-gateways (AWS CLI)
- ec2-describe-internet-gateways (Amazon EC2 CLI)
- Get-EC2InternetGateway (AWS Tools for Windows PowerShell)

Detach an Internet gateway from a VPC

- detach-internet-gateway (AWS CLI)
- ec2-detach-internet-gateway (Amazon EC2 CLI)
- Dismount-EC2InternetGateway (AWS Tools for Windows PowerShell)

Delete an Internet gateway

- delete-internet-gateway (AWS CLI)
- ec2-delete-internet-gateway (Amazon EC2 CLI)
- Remove-EC2InternetGateway (AWS Tools for Windows PowerShell)

NAT Instances

Instances that you launch into a private subnet in a virtual private cloud (VPC) can't communicate with the Internet. You can optionally use a network address translation (NAT) instance in a public subnet in your VPC to enable instances in the private subnet to initiate outbound traffic to the Internet, but prevent the instances from receiving inbound traffic initiated by someone on the Internet.

For a general overview of VPCs and subnets, see What is Amazon VPC? (p. 1). For more information about public and private subnets, see Subnet Routing (p. 53).

> **Note**
> We use the term *NAT instance*; however, the primary role of a NAT instance is actually port address translation (PAT). We chose to use the more widely known term, NAT. For more information about NAT and PAT, see the Wikipedia article about network address translation.

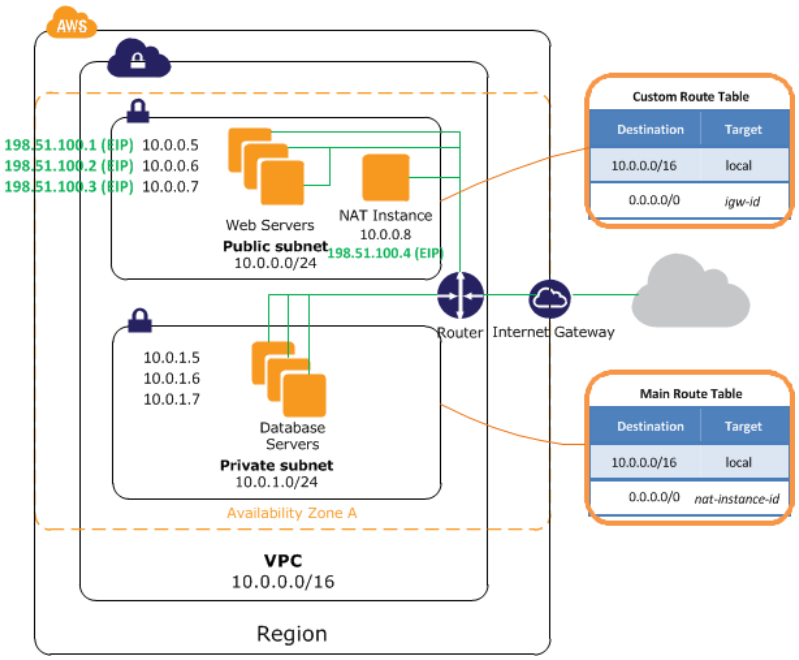
Topics

- NAT Instance Basics (p. 139)
- Setting up the NAT Instance (p. 139)
- Creating the NATSG Security Group (p. 141)
- Disabling Source/Destination Checks (p. 142)

- Updating the Main Route Table (p. 143)
- Testing Your NAT Instance Configuration (p. 143)

NAT Instance Basics

The following figure illustrates the NAT instance basics. The main route table sends the traffic from the instances in the private subnet to the NAT instance in the public subnet. The NAT instance sends the traffic to the Internet gateway for the VPC. The traffic is attributed to the Elastic IP address of the NAT instance. The NAT instance specifies a high port number for the response; if a response comes back, the NAT instance sends it to an instance in the private subnet based on the port number for the response.



Setting up the NAT Instance

You can use the VPC wizard to set up a VPC with a NAT instance; for more information, see Scenario 2: VPC with Public and Private Subnets (NAT) (p. 22). The wizard performs many of the configuration steps for you, including launching a NAT instance, and setting up the routing. However, if you prefer, you can create and configure a VPC and a NAT instance manually using the steps below.

1. Create a VPC with two subnets.

   Note
   The steps below are for manually creating and configuring a VPC; not for creating a VPC using the VPC wizard.

   a. Create a VPC (see Creating a VPC (p. 49))
   b. Create two subnets (see Creating a Subnet (p. 135))
   c. Attach an Internet gateway to the VPC (see Attaching an Internet Gateway (p. 135))
   d. Create a custom route table that sends traffic destined outside the VPC to the Internet gateway, and then associate it with one subnet, making it a public subnet (see Creating a Custom Route Table (p. 135))

2. Create the NATSG security group (see Creating the NATSG Security Group (p. 141)). You'll specify this security group when you launch the NAT instance.
3. Launch an instance into your public subnet from an AMI that's been configured to run as a NAT instance. Amazon provides Amazon Linux AMIs that are configured to run as NAT instances. These AMIs include the string `amzn-ami-vpc-nat` in their names, so you can search for them in the Amazon EC2 console.

   a. Open the Amazon EC2 console.
   b. On the dashboard, click the **Launch Instance** button, and complete the wizard as follows:

      i. On the **Choose an Amazon Machine Image (AMI)** page, select the **Community AMIs** category, and search for `amzn-ami-vpc-nat`. In the results list, each AMI's name includes the version to enable you to select the most recent AMI, for example, `2013.09`. Click **Select**.
      ii. On the **Choose an Instance Type** page, select the instance type, then click **Next: Configure Instance Details**.
      iii. On the **Configure Instance Details** page, select the VPC you created from the **Network** list, and select your public subnet from the **Subnet** list.
      iv. (Optional) Select the **Public IP** check box to request that your NAT instance receives a public IP address. If you choose not to assign a public IP address now, you can allocate an Elastic IP address and assign it to your instance after it's launched. For more information about assigning a public IP at launch, see Assigning a Public IP Address During Launch (p. 118). Click **Next: Add Storage**.
      v. You can choose to add storage to your instance, and on the next page, you can add tags. Click **Next: Configure Security Group** when you are done.
      vi. On the **Configure Security Group** page, select the **Select an existing security group** option, and select the NATSG security group that you created. Click **Review and Launch**.
      vii. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to choose a key pair and launch your instance.

4. (Optional) Connect to the NAT instance, make any modifications that you need, and then create your own AMI that's configured to run as a NAT instance. You can use this AMI the next time that you need to launch a NAT instance. For more information about creating an AMI, see Creating Amazon EBS-Backed AMIs in the *Amazon EC2 User Guide for Linux Instances*.
5. Disable the `SrcDestCheck` attribute for the NAT instance (see Disabling Source/Destination Checks (p. 142))
6. If you did not assign a public IP address to your NAT instance during launch (step 3), you need to associate an Elastic IP address with it.

   a. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
   b. Click **Elastic IPs** in the navigation pane.
   c. Click the **Allocate New Address** button.
   d. In the **Allocate New Address** dialog box, in the **Network platform** list, select **EC2-VPC**, and then click **Yes, Allocate**.
   e. Select the Elastic IP address from the list, and then click the **Associate Address** button.
   f. In the **Associate Address** dialog box, select the network interface for the NAT instance. Select the address to associate the EIP with from the **Private IP address** list, and then click **Yes, Associate**.

7. Update the main route table to send traffic to the NAT instance. For more information, see Updating the Main Route Table (p. 143).

Launching a NAT Instance Using the Command Line

To launch a NAT instance into your subnet, use one of the following commands. For more information about these command line interfaces, see Accessing Amazon VPC (p. 6).

- run-instances (AWS CLI)
- ec2-run-instances (Amazon EC2 CLI)
- New-EC2Instance (AWS Tools for Windows PowerShell)

To get the ID of an AMI that's configured to run as a NAT instance, use a command to describe images, and use filters to return results only for AMIs that are owned by Amazon, and that have the `amzn-ami-vpc-nat` string in their names. The following example uses the AWS CLI:

```
PROMPT> aws ec2 describe-images --filter Name="owner-alias",Values="amazon" --
filter Name="name",Values="amzn-ami-vpc-nat*"
```

Creating the NATSG Security Group

Define the NATSG security group as described in the following table to enable your NAT instance to receive Internet-bound traffic from instances in a private subnet, as well as SSH traffic from your network. The NAT instance can also send traffic to the Internet, which enables the instances in the private subnet to get software updates.

NATSG: Recommended Rules

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
`10.0.1.0/24`	TCP	80	Allow inbound HTTP traffic from servers in the private subnet
`10.0.1.0/24`	TCP	443	Allow inbound HTTPS traffic from servers in the private subnet
Public IP address range of your home network	TCP	22	Allow inbound SSH access to the NAT instance from your home network (over the Internet gateway)
Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
`0.0.0.0/0`	TCP	80	Allow outbound HTTP access to the Internet
`0.0.0.0/0`	TCP	443	Allow outbound HTTPS access to the Internet

To create the NATSG security group

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. Click **Security Groups** in the navigation pane.
3. Click the **Create Security Group** button.

4. In the **Create Security Group** dialog box, specify `NATSG` as the name of the security group, and provide a description. Select the ID of your VPC from the **VPC** list, and then click **Yes, Create**.
5. Select the NATSG security group that you just created. The details pane displays the details for the security group, plus tabs for working with its inbound and outbound rules.
6. Add rules for inbound traffic using the **Inbound Rules** tab as follows:
   a. Click **Edit**.
   b. Click **Add another rule**, and select **HTTP** from the **Type** list. In the **Source** field, specify the IP address range of your private subnet.
   c. Click **Add another rule**, and select **HTTPS** from the **Type** list. In the **Source** field, specify the IP address range of your private subnet.
   d. Click **Add another rule**, and select **SSH** from the **Type** list. In the **Source** field, specify the public IP address range of your network.
   e. Click **Save**.
7. Add rules for outbound traffic using the **Outbound Rules** tab as follows:
   a. Click **Edit**.
   b. Click **Add another rule**, and select **HTTP** from the **Type** list. In the **Destination** field, specify `0.0.0.0/0`
   c. Click **Add another rule**, and select **HTTPS** from the **Type** list. In the **Destination** field, specify `0.0.0.0/0`
   d. Click **Save**.

For more information about security groups, see Security Groups for Your VPC (p. 64).

Disabling Source/Destination Checks

Each EC2 instance performs source/destination checks by default. This means that the instance must be the source or destination of any traffic it sends or receives. However, a NAT instance must be able to send and receive traffic when the source or destination is not itself. Therefore, you must disable source/destination checks on the NAT instance.

You can disable the `SrcDestCheck` attribute for a NAT instance that's either running or stopped using the console or the command line.

To disable source/destination checking using the console

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, click **Instances**.
3. Select the NAT instance, click **Actions**, select **Networking**, and then select **Change Source/Dest. Check**.
4. For the NAT instance, verify that this attribute is disabled. Otherwise, click **Yes, Disable**.

To disable source/destination checking using the command line

You can use one of the following commands. For more information about these command line interfaces, see Accessing Amazon VPC (p. 6).

- modify-instance-attribute (AWS CLI)
- ec2-modify-instance-attribute (Amazon EC2 CLI)

- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

Updating the Main Route Table

Update the main route table as described in the following procedure. By default, the main route table enables the instances in your VPC to communicate with each other. We'll add a route that sends all other subnet traffic to the NAT instance.

To update the main route table

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**.
3. Select the main route table for your VPC. The details pane displays tabs for working with its routes, associations, and route propagation.
4. On the **Routes** tab, click **Edit**, specify `0.0.0.0/0` in the **Destination** box, select the instance ID of the NAT instance from the **Target** list, and then click **Save**.
5. On the **Subnet Associations** tab, click **Edit**, and then select the **Associate** check box for the subnet. Click **Save**.

For more information about route tables, see Route Tables (p. 123).

Testing Your NAT Instance Configuration

After you have launched a NAT instance and completed the configuration steps above, you can perform a test to check if an instance in your private subnet can access the Internet through the NAT instance. To do this, update your NAT instance's security group rules to allow inbound and outbound ICMP traffic and allow outbound SSH traffic, launch an instance into your private subnet, configure SSH agent forwarding to access instances in your private subnet, connect to your instance, and then test the Internet connectivity.

To update your NAT instance's security group

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, click **Security Groups**.
3. Find the security group associated with your NAT instance, and click **Edit** in the **Inbound** tab.
4. Click **Add Rule**, select **All ICMP** from the **Type** list, and select **Custom IP** from the **Source** list. Enter the IP address range of your private subnet, for example, `10.0.1.0/24`. Click **Save**.
5. In the **Outbound** tab, click **Edit**.
6. Click **Add Rule**, select **SSH** from the **Type** list, and select **Custom IP** from the **Source** list. Enter the IP address range of your private subnet, for example, `10.0.1.0/24`. Click **Save**.
7. Click **Add Rule**, select **All ICMP** from the **Type** list, and select **Custom IP** from the **Source** list. Enter `0.0.0.0/0`, and then click **Save**.

To launch an instance into your private subnet

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, click **Instances**.
3. Launch an instance into your private subnet. For more information, see Launching an Instance into Your Subnet (p. 54). Ensure that you configure the following options in the launch wizard, and then click **Launch**:

- On the **Choose an Amazon Machine Image (AMI)** page, select an Amazon Linux AMI from the **Quick Start** category.
- On the **Configure Instance Details** page, select your private subnet from the **Subnet** list, and do not assign a public IP address to your instance.
- On the **Configure Security Group** page, ensure that your security group includes an inbound rule that allows SSH access from your NAT instance's private IP address, or from the IP address range of your public subnet, and ensure that you have an outbound rule that allows outbound ICMP traffic.
- In the **Select an existing key pair or create a new key pair** dialog box, select the same key pair you used to launch the NAT instance.

To configure SSH agent forwarding for Linux or OS X

1. From your local machine, add your private key to the authentication agent.

   For Linux, use the following command:

   ```
   PROMPT> ssh-add -c mykeypair.pem
   ```

   For OS X, use the following command:

   ```
   PROMPT> ssh-add -K mykeypair.pem
   ```

2. Connect to your NAT instance using the `-A` option to enable SSH agent forwarding, for example:

   ```
   ssh -A ec2-user@54.0.0.123
   ```

To configure SSH agent forwarding for Windows (PuTTY)

1. Download and install Pageant from the PuTTY download page, if not already installed.
2. Convert your private key to .ppk format. For more information, see Converting Your Private Key Using PuTTYgen.
3. Start Pageant, and then click **Add Key**. Select the .ppk file you created, enter the passphrase if required, click **OK**, and then close the **Pageant Key List** window.
4. Start a PuTTY session to connect to your NAT instance. In the **Auth** category, ensure that you select the **Allow agent forwarding** option, and leave the **Private key file for authentication** field blank.

To test the Internet connection

1. Test that your NAT instance can communicate with the Internet by running the `ping` command for a website that has ICMP enabled; for example:

   ```
   PROMPT> ping ietf.org

   PING ietf.org (4.31.198.44) 56(84) bytes of data.
   64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=1 ttl=48 time=74.9 ms
   64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=2 ttl=48 time=75.1 ms
   ...
   ```

Press **Ctrl+C** on your keyboard to cancel the `ping` command.

2. From your NAT instance, connect to your instance in your private subnet by using its private IP address, for example:

```
PROMPT> ssh ec2-user@10.0.1.123
```

3. From your private instance, test that you can connect to the Internet by running the `ping` command:

```
PROMPT> ping ietf.org

PING ietf.org (4.31.198.44) 56(84) bytes of data.
64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=1 ttl=47 time=86.0 ms
64 bytes from mail.ietf.org (4.31.198.44): icmp_seq=2 ttl=47 time=75.6 ms
...
```

Press **Ctrl+C** on your keyboard to cancel the `ping` command.

If the `ping` command fails, check the following information:

- Check that your NAT instance's security group rules allow inbound ICMP traffic from your private subnet. If not, your NAT instance cannot receive the `ping` command from your private instance.
- Check that you've configured your route tables correctly. For more information, see Updating the Main Route Table (p. 143).
- Ensure that you've disabled source/destination checking for your NAT instance. For more information, see Disabling Source/Destination Checks (p. 142).
- Ensure that you are pinging a website that has ICMP enabled. If not, you will not receive reply packets. To test this, perform the same `ping` command from the command line terminal on your own computer.

4. (Optional) Terminate your private instance if you no longer require it. For more information, see Terminate Your Instance in the *Amazon EC2 User Guide for Linux Instances*.

DHCP Options Sets

This topic describes DHCP options sets and how to specify the DHCP options for your VPC.

Topics

- Overview of DHCP Options Sets (p. 145)
- Amazon DNS Server (p. 146)
- Changing DHCP Options (p. 147)
- Working with DHCP Options Sets (p. 147)
- API and Command Overview (p. 149)

Overview of DHCP Options Sets

The Dynamic Host Configuration Protocol (DHCP) provides a standard for passing configuration information to hosts on a TCP/IP network. The *`options`* field of a DHCP message contains the configuration parameters. Some of those parameters are the domain name, domain name server, and the netbios-node-type.

DHCP options sets are associated with your AWS account so that you can use them across all of your virtual private clouds (VPC).

The Amazon EC2 instances you launch into a nondefault VPC are private by default; they're not assigned a public IP address unless you specifically assign one during launch, or you modify the subnet's public IP address attribute. By default, all instances in a nondefault VPC receive an unresolvable host name that AWS assigns (for example, ip-10-0-0-202). You can assign your own domain name to your instances, and use up to four of your own DNS servers. To do that, you must specify a special set of DHCP options to use with the VPC.

The following table lists all the supported options for a DHCP options set. You can specify only the options you need in your DHCP options set. For more information about the options, see RFC 2132.

DHCP Option Name	Description
domain-name-servers	The IP addresses of up to four domain name servers, or AmazonProvidedDNS. The default DHCP option set specifies AmazonProvidedDNS. If specifying more than one domain name server, separate them with commas.
domain-name	If you're using AmazonProvidedDNS in `us-east-1`, specify `ec2.internal`. If you're using AmazonProvidedDNS in another region, specify *region*.compute.internal (for example, `ap-northeast-1.compute.internal`). Otherwise, specify a domain name (for example, `MyCompany.com`). **Important** Some Linux operating systems accept multiple domain names separated by spaces. However, other Linux operating systems and Windows treat the value as a single domain, which results in unexpected behavior. If your DHCP options set is associated with a VPC that has instances with multiple operating systems, specify only one domain name.
ntp-servers	The IP addresses of up to four Network Time Protocol (NTP) servers. For more information, see section 8.3 of RFC 2132.
netbios-name-servers	The IP addresses of up to four NetBIOS name servers.
netbios-node-type	The NetBIOS node type (1, 2, 4, or 8). We recommend that you specify 2 (broadcast and multicast are not currently supported). For more information about these node types, see RFC 2132.

Amazon DNS Server

When you create a VPC, we automatically create a set of DHCP options and associate them with the VPC. This set includes two options: `domain-name-servers=AmazonProvidedDNS`, and `domain-name=`*`domain-name-for-your-region`*. AmazonProvidedDNS is an Amazon DNS server,

and this option enables DNS for instances that need to communicate over the VPC's Internet gateway. The string `AmazonProvidedDNS` maps to a DNS server running on a reserved IP address at the base of the VPC network range "plus two". For example, the DNS Server on a 10.0.0.0/16 network is located at 10.0.0.2.

The Amazon DNS server in your VPC is used to resolve the DNS domain names that you specify in a private hosted zone in Amazon Route 53. For more information about private hosted zones, see Working with Private Hosted Zones in the *Amazon Route 53 Developer Guide*.

Services that use the Hadoop framework, such as Amazon EMR, require instances to resolve their own fully qualified domain names (FQDN). In such cases, DNS resolution can fail if the `domain-name-servers` option is set to a custom value. To ensure proper DNS resolution, consider adding a conditional forwarder on your DNS server to forward queries for the domain *`region-name`*`.compute.internal` to the Amazon DNS server. For more information about launching an Amazon EMR cluster into a VPC, see Setting Up a VPC to Host Clusters in the *Amazon Elastic MapReduce Developer Guide*.

Note
You can use the Amazon DNS server IP address 169.254.169.253, though some servers don't allow its use. Windows Server 2008, for example, disallows the use of a DNS server located in the 169.254.x.x network range.

Changing DHCP Options

After you create a set of DHCP options, you can't modify them. If you want your VPC to use a different set of DHCP options, you must create a new set and associate them with your VPC. You can also set up your VPC to use no DHCP options at all.

You can have multiple sets of DHCP options, but you can associate only one set of DHCP options with a VPC at a time. If you delete a VPC, the DHCP options set associated with the VPC are also deleted.

After you associate a new set of DHCP options with a VPC, any existing instances and all new instances that you launch in the VPC use these options. You don't need to restart or relaunch the instances. They automatically pick up the changes within a few hours, depending on how frequently the instance renews its DHCP lease. If you want, you can explicitly renew the lease using the operating system on the instance.

Working with DHCP Options Sets

This section shows you how to work with DHCP options sets.

Topics

- Creating a DHCP Options Set (p. 147)
- Changing the Set of DHCP Options a VPC Uses (p. 148)
- Changing a VPC to use No DHCP Options (p. 148)
- Deleting a DHCP Options Set (p. 149)

Creating a DHCP Options Set

You can create as many additional DHCP options sets as you want. However, you can only associate a VPC with one set of DHCP options at a time. After you create a set of DHCP options, you must configure your VPC to use it. For more information, see Changing the Set of DHCP Options a VPC Uses (p. 148).

To create a DHCP options set

1. Open the Amazon VPC console.

2. Click **DHCP Options Sets** in the navigation pane, and then click the **Create DHCP Options Set** button.
3. In the **Create DHCP Options Set** dialog box, enter values for the options that you want to use, and then click **Yes, Create**.

   Important
   If your VPC has an Internet gateway, make sure to specify your own DNS server or Amazon's DNS server (AmazonProvidedDNS) for the **Domain name servers** value. Otherwise, the instances that need to communicate with the Internet won't have access to DNS.

   The new set of DHCP options appears in your list of DHCP options.
4. Make a note of the ID of the new set of DHCP options (dopt-*xxxxxxxx*). You will need it to associate the new set of options with your VPC.

Although you've created a set of DHCP options, you must associate it with your VPC for the options to take effect. You can create multiple sets of DHCP options, but you can associate only one set of DHCP options with your VPC at a time.

Changing the Set of DHCP Options a VPC Uses

You can change which set of DHCP options your VPC uses. If you want the VPC to use no DHCP options, see Changing a VPC to use No DHCP Options (p. 148).

Note
The following procedure assumes that you've already created the DHCP options set you want to change to. If you haven't, create the options set now. For more information, see Creating a DHCP Options Set (p. 147).

To change the DHCP options set associated with a VPC

1. Open the Amazon VPC console.
2. Click **Your VPCs** in the navigation pane.
3. Select the VPC, and select **Edit DHCP Options Set** from the **Actions** list.
4. In the **DHCP options set** list, select a set of options from the list, and then click **Save**.

After you associate a new set of DHCP options with the VPC, any existing instances and all new instances that you launch in that VPC use the options. You don't need to restart or relaunch the instances. They automatically pick up the changes within a few hours, depending on how frequently the instance renews its DHCP lease. If you want, you can explicitly renew the lease using the operating system on the instance.

Changing a VPC to use No DHCP Options

You can set up your VPC to use no set of DHCP options.

1. Open the Amazon VPC console.
2. Click **Your VPCs** in the navigation pane.
3. Select the VPC, and select **default** from the **Actions** list.
4. In the **DHCP options set** list, select **default** from the list, and then click **Save**.

You don't need to restart or relaunch the instances. They automatically pick up the changes within a few hours, depending on how frequently the instance renews its DHCP lease. If you want, you can explicitly renew the lease using the operating system on the instance.

Deleting a DHCP Options Set

When you no longer need a DHCP options set, use the following procedure to delete it. The VPC must not be using the set of options.

To delete a DHCP options set

1. Open the Amazon VPC console.
2. Click **DHCP Options Sets** in the navigation pane.
3. Select the set of DHCP options to delete, and then click **Delete**.
4. In the **Delete DHCP Options Set** dialog box, click **Yes, Delete**.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Create a set of DHCP options for your VPC

- create-dhcp-options (AWS CLI)
- ec2-create-dhcp-options (Amazon EC2 CLI)
- New-EC2DhcpOption (AWS Tools for Windows PowerShell)

Associate a set of DHCP options with the specified VPC, or no DHCP options

- associate-dhcp-options (AWS CLI)
- ec2-associate-dhcp-options (Amazon EC2 CLI)
- Register-EC2DhcpOption (AWS Tools for Windows PowerShell)

Describes one or more sets of DHCP options

- describe-dhcp-options (AWS CLI)
- ec2-describe-dhcp-options (Amazon EC2 CLI)
- Get-EC2DhcpOption (AWS Tools for Windows PowerShell)

Deletes a set of DHCP options

- delete-dhcp-options (AWS CLI)
- ec2-delete-dhcp-options (Amazon EC2 CLI)
- Remove-EC2DhcpOption (AWS Tools for Windows PowerShell)

Using DNS with Your VPC

Amazon EC2 instances need IP addresses to communicate. Public IP addresses enable communication over the Internet, while private IP addresses enable communication within the network of the instance (either EC2-Classic or a VPC).

Domain Name System (DNS) is a standard by which names used on the Internet are resolved to their corresponding IP addresses. A DNS hostname is a name that uniquely and absolutely names a computer;

it's composed of a host name and a domain name. DNS servers resolve DNS hostnames to their corresponding IP addresses.

We provide an Amazon DNS server. To use your own DNS server, update the DHCP options set for your VPC. For more information, see DHCP Options Sets (p. 145).

To enable an EC2 instance to be publicly accessible, it must have a public IP address, a DNS hostname, and DNS resolution.

Topics

- Viewing DNS Hostnames for Your EC2 Instance (p. 150)
- Updating DNS Support for Your VPC (p. 151)
- Using Private Hosted Zones (p. 152)

Viewing DNS Hostnames for Your EC2 Instance

When you launch an instance into the EC2-Classic platform or into a default VPC, we provide the instance with public and private DNS hostnames. Instances that you launch into a nondefault VPC might have public and private DNS hostnames, depending on the settings you specify for the VPC and for the instance.

You can view the DNS hostnames for a running instance or a network interface using the Amazon EC2 console or the command line.

Instance

To view DNS hostnames for an instance using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Instances**.
3. Select the instance from the list.
4. In the **Description** tab in the details pane, review the values of the **Public DNS** and **Private DNS** fields.

To view DNS hostnames for an instance using the command line

You can use one of the following commands. For more information about these command line interfaces, see Accessing Amazon VPC (p. 6).

- describe-instances (AWS CLI)
- ec2-describe-instances (Amazon EC2 CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

Network Interface

To view DNS hostnames for a network interface using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Network Interfaces**.
3. Select the network interface from the list.
4. In the **Details** tab for the network interface, review the values of the **Public DNS** and **Private DNS** fields.

To view DNS hostnames for a network interface using the command line

You can use one of the following commands. For more information about these command line interfaces, see Accessing Amazon VPC (p. 6).

- describe-network-interfaces (AWS CLI)
- ec2-describe-network-interfaces (Amazon EC2 CLI)
- Get-EC2NetworkInterface (AWS Tools for Windows PowerShell)

Updating DNS Support for Your VPC

When you launch an instance into a VPC, we provide the instance with public and private DNS hostnames only if DNS hostnames are enabled for the VPC. By default, DNS hostnames are enabled only for default VPCs and VPCs that you create using the VPC wizard in the VPC console.

We support the following VPC attributes to control DNS support. Be sure to set both attributes to `true` if you want your instances to have public DNS hostnames that are accessible from the Internet.

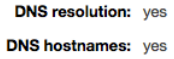
Attribute	Description
`enableDnsHostnames`	Indicates whether the instances launched in the VPC get DNS hostnames. If this attribute is `true`, instances in the VPC get DNS hostnames; otherwise, they do not. If you want your instances to get DNS hostnames, you must also set the `enableDnsSupport` attribute to `true`.
`enableDnsSupport`	Indicates whether the DNS resolution is supported for the VPC. If this attribute is `false`, the Amazon provided DNS service in the VPC that resolves public DNS hostnames to IP addresses is not enabled. If this attribute is `true`, queries to the Amazon provided DNS server at the 169.254.169.253 IP address, or the reserved IP address at the base of the VPC network range "plus two" will succeed. For more information, see Amazon DNS Server (p. 146).

The Amazon DNS server cannot resolve private DNS hostnames if your VPC's IP address range falls outside of the private IP addresses ranges specified by RFC 1918.

If you enable DNS hostnames and DNS support in a VPC that didn't previously support them, an instance that you already launched into that VPC gets a public DNS hostname if it has a public IP address or an Elastic IP address.

To describe and update DNS support for a VPC using the console

1. Open the Amazon VPC console.
2. In the navigation pane, click **Your VPCs**.
3. Select the VPC from the list.
4. Review the information in the **Summary** tab. In this example, both settings are enabled.



5. To update these settings, click the **Actions** list, and select **Edit DNS Resolution** or **Edit DNS Hostnames**. In the dialog box that opens, select **Yes** or **No**, and then click **Save**.

To describe DNS support for a VPC using the command line

You can use one of the following commands. For more information about these command line interfaces, see Accessing Amazon VPC (p. 6).

- describe-vpc-attribute (AWS CLI)
- ec2-describe-vpc-attribute (Amazon EC2 CLI)
- Get-EC2VpcAttribute (AWS Tools for Windows PowerShell)

To update DNS support for a VPC using the command line

You can use one of the following commands. For more information about these command line interfaces, see Accessing Amazon VPC (p. 6).

- modify-vpc-attribute (AWS CLI)
- ec2-modify-vpc-attribute (Amazon EC2 CLI)
- Edit-EC2VpcAttribute (AWS Tools for Windows PowerShell)

Using Private Hosted Zones

If you want to access the resources in your VPC using custom DNS domain names, such as example.com, instead of using private IP addresses or AWS-provided private DNS hostnames, you can create a private hosted zone in Amazon Route 53. A private hosted zone is a container that holds information about how you want to route traffic for a domain and its subdomains within one or more VPCs without exposing your resources to the Internet. You can then create Amazon Route 53 resource record sets, which determine how Amazon Route 53 responds to queries for your domain and subdomains. For example, if you want browser requests for example.com to be routed to a web server in your VPC, you'll create an A record in your private hosted zone and specify the IP address of that web server. For more information about creating a private hosted zone, see Working with Private Hosted Zones in the *Amazon Route 53 Developer Guide*.

To access resources using custom DNS domain names, you must be connected to an instance within your VPC. From your instance, you can test that your resource in your private hosted zone is accessible from its custom DNS name by using the `ping` command; for example, `ping mywebserver.example.com`. (You must ensure that your instance's security group rules allow inbound ICMP traffic for the `ping` command to work.)

Private hosted zones do not support transitive relationships outside of the VPC; for example, you cannot access your resources using their custom private DNS names from the other side of a VPN connection.

> **Important**
> If you are using custom DNS domain names defined in a private hosted zone in Amazon Route 53, you must set the following VPC attributes to `true`: `enableDnsHostnames` and `enableDnsSupport`. For more information, see Updating DNS Support for Your VPC (p. 151).

VPC Peering

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IP addresses. Instances in either VPC can communicate with each other as

if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account within a single region.

AWS uses the existing infrastructure of a VPC to create a VPC peering connection; it is neither a gateway nor a VPN connection, and does not rely on a separate piece of physical hardware. There is no single point of failure for communication or a bandwidth bottleneck.

A VPC peering connection can help you to facilitate the transfer of data; for example, if you have more than one AWS account, you can peer the VPCs across those accounts to create a file sharing network. You can also use a VPC peering connection to allow other VPCs to access resources you have in one of your VPCs. For more examples of scenarios in which you can use a VPC peering connection, see the Amazon VPC Peering Guide.

Topics

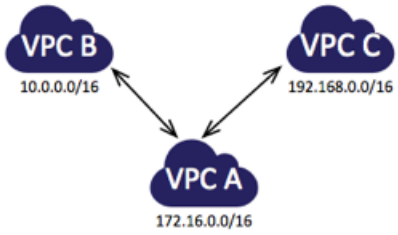
- VPC Peering Basics (p. 153)
- Working with VPC Peering Connections (p. 155)
- API and CLI Overview (p. 159)
- Controlling Access to VPC Peering Connections (p. 160)

VPC Peering Basics

To establish a VPC peering connection, the owner of the *requester VPC* (or *local VPC*) sends a request to the owner of the *peer VPC* to create the VPC peering connection. The peer VPC can be owned by you, or another AWS account, and cannot have a CIDR block that overlaps with the requester VPC's CIDR block. The owner of the peer VPC has to accept the VPC peering connection request to activate the VPC peering connection. To enable the flow of traffic between the peer VPCs using private IP addresses, add a route to one or more of your VPC's route tables that points to the IP address range of the peer VPC. The owner of the peer VPC adds a route to one of their VPC's route tables that points to the IP address range of your VPC. You may also need to update the security group rules that are associated with your instance to ensure that traffic to and from the peer VPC is not restricted. For more information about security groups, see Security Groups for Your VPC (p. 64).

A VPC peering connection is a one to one relationship between two VPCs. You can create multiple VPC peering connections for each VPC that you own, but transitive peering relationships are not supported: you will not have any peering relationship with VPCs that your VPC is not directly peered with.

The following diagram is an example of one VPC peered to two different VPCs. There are two VPC peering connections: VPC A is peered with both VPC B and VPC C. VPC B and VPC C are not peered, and you cannot use VPC A as a transit point for peering between VPC B and VPC C. If you want to enable routing of traffic between VPC B and VPC C, you must create a unique VPC peering connection between them.

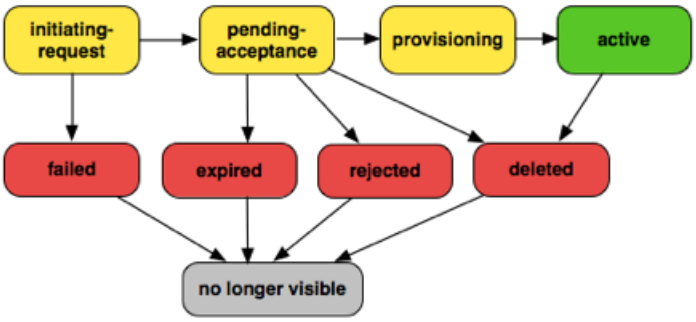


For more information about VPC peering limitations, see VPC Peering Limitations (p. 154). For more information and examples of peering relationships that are supported, see the *Amazon VPC Peering Guide*.

You are charged for data transfer within a VPC peering connection at the same rate as you are charged for data transfer across Availability Zones. For more information, see Amazon EC2 Pricing.

VPC Peering Connection Lifecycle

A VPC peering connection goes through various stages starting from when the request is initiated. At each stage, there may be actions that you can take, and at the end of its lifecycle, the VPC peering connection remains visible in the VPC console and API or command line output for a period of time.



- **Initiating-request**: A request for a VPC peering connection has been initiated. At this stage, the peering connection may fail or may go to `pending-acceptance`.
- **Failed**: The request for the VPC peering connection has failed. During this state, it cannot be accepted or rejected. The failed VPC peering connection remains visible to the requester for 2 hours.
- **Pending-acceptance**: The VPC peering connection request is awaiting acceptance from the owner of the peer VPC. During this state, the owner of the requester VPC can delete the request, and the owner of the peer VPC can accept or reject the request. If no action is taken on the request, it will expire after 7 days.
- **Expired**: The VPC peering connection request has expired, and no action can be taken on it by either VPC owner. The expired VPC peering connection remains visible to both VPC owners for 2 days.
- **Rejected**: The owner of the peer VPC has rejected a `pending-acceptance` VPC peering connection request. During this state, the request cannot be accepted. The rejected VPC peering connection remains visible to the owner of the requester VPC for 2 days, and visible to the owner of the peer VPC for 2 hours. If the request was created within the same AWS account, the rejected request remains visible for 2 hours.
- **Provisioning**: The VPC peering connection request has been accepted, and will soon be in the `active` state.
- **Active**: The VPC peering connection is active. During this state, either of the VPC owners can delete the VPC peering connection, but cannot reject it.
- **Deleted**: An `active` VPC peering connection has been deleted by either of the VPC owners, or a `pending-acceptance` VPC peering connection request has been deleted by the owner of the requester VPC. During this state, the VPC peering connection cannot be accepted or rejected. The VPC peering connection remains visible to the party that deleted it for 2 hours, and visible to the other party for 2 days. If the VPC peering connection was created within the same AWS account, the deleted request remains visible for 2 hours.

VPC Peering Limitations

To create a VPC peering connection with another VPC, you need to be aware of the following limitations and rules:

- You cannot create a VPC peering connection between VPCs that have matching or overlapping CIDR blocks.

- You cannot create a VPC peering connection between VPCs in different regions.
- You have a limit on the number active and pending VPC peering connections that you can have per VPC. For more information about VPC limits, see Amazon VPC Limits (p. 194).
- VPC peering does not support transitive peering relationships; in a VPC peering connection, your VPC will not have access to any other VPCs that the peer VPC may be peered with. This includes VPC peering connections that are established entirely within your own AWS account. For more information and examples of peering relationships that are supported, see the *Amazon VPC Peering Guide*.
- You cannot have more than one VPC peering connection between the same two VPCs at the same time.
- The Maximum Transmission Unit (MTU) across a VPC peering connection is 1500 bytes.
- A placement group can span peered VPCs; however, you will not get full-bisection bandwidth between instances in peered VPCs. For more information about placement groups, see Placement Groups in the *Amazon EC2 User Guide for Linux Instances*.
- Unicast reverse path forwarding in VPC peering connections is not supported. For more information, see Routing for Response Traffic in the *Amazon VPC Peering Guide*.
- You cannot reference a security group from the peer VPC as a source or destination for ingress or egress rules in your security group. Instead, reference CIDR blocks of the peer VPC as the source or destination of your security group's ingress or egress rules.
- Private DNS values cannot be resolved between instances in peered VPCs.

Working with VPC Peering Connections

Topics

- Creating a VPC Peering Connection (p. 155)
- Accepting a VPC Peering Connection (p. 157)
- Rejecting a VPC Peering Connection (p. 157)
- Updating Route Tables for Your VPC Peering Connection (p. 157)
- Describing Your VPC Peering Connections (p. 158)
- Deleting a VPC Peering Connection (p. 159)

Creating a VPC Peering Connection

To create a VPC peering connection, first create a request to peer with another VPC. You can request a VPC peering connection with another VPC in your account, or with a VPC in a different AWS account. To activate the request, the owner of the peer VPC must accept the request.

Creating a VPC Peering Connection with Another VPC in Your Account

To request a VPC peering connection with a VPC in your account, ensure that you have the IDs of the VPCs with which you are creating the VPC peering connection. You must both create and accept the VPC peering connection request yourself to activate it.

To create a VPC peering connection in your account

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Peering Connections**.
3. Click **Create VPC Peering Connection**.
4. In the dialog, configure the following information, and click **Create** when you are done:
   - **Name**: You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify.

- **Local VPC to peer**: Select the VPC in your account with which you want to create the VPC peering connection.
- **Select a VPC to peer with**: Ensure **My account** is selected, and select another of your VPCs from the **VPC ID** list. Only VPCs in the current region are displayed.

  Important
  Ensure that your VPCs do not have overlapping CIDR blocks. If they do, the status of the VPC peering connection immediately goes to `failed`.

5. A confirmation dialog box provides the ID of the VPC peering connection, as well as information about the VPCs in the peering connection. Click **OK**.
6. To view all VPC peering connections that are pending your acceptance, select **Requests pending my approval** from the filter list.
7. Select the VPC peering connection that you've created, and click **Accept request**.
8. In the confirmation dialog, click **Yes, Accept**. A second confirmation dialog displays; click **Modify my route tables now** to go directly to the route tables page, or click **Close** to do this later.

Now that your VPC peering connection is active, you must add an entry to your VPCs' route tables to enable traffic to be directed between the peered VPCs. For more information, see Updating Route Tables for Your VPC Peering Connection (p. 157).

Creating a VPC Peering Connection with a VPC in Another AWS Account

You can request a VPC peering connection with a VPC that's in another AWS account. Before you begin, ensure that you have the AWS account number and VPC ID of the VPC to peer with. After you've created the request, the owner of the peer VPC must accept the VPC peering connection to activate it.

To create a VPC peering connection with a remote VPC

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Peering Connections**.
3. Click **Create VPC Peering Connection**.
4. In the dialog, configure the information as follows, and click **Create** when you are done:

   - **Name**: You can optionally name your VPC peering connection. Doing so creates a tag with a key of `Name` and a value that you specify. This tag is only visible to you; the owner of the peer VPC can create their own tags for the VPC peering connection.
   - **Local VPC to peer**: Select the VPC in your account with which to create the VPC peering connection.
   - **Select a VPC to peer with**: Select **Another account**, and enter the AWS account ID and the ID of the VPC with which to create the VPC peering connection.

     Important
     If your VPC and the peer VPC have overlapping CIDR blocks, or if the account ID and VPC ID are incorrect or do not correspond with each other, the status of the VPC peering connection immediately goes to `failed`.

5. A confirmation dialog box provides the ID of the VPC peering connection, as well as information about the VPCs in the peering connection. Click **OK**.

The VPC peering connection that you've created is not active. To activate it, the owner of the peer VPC must accept the VPC peering connection request. To enable traffic to be directed to the peer VPC, update

your VPC's route table. For more information, see Updating Route Tables for Your VPC Peering Connection (p. 157).

Accepting a VPC Peering Connection

A VPC peering connection that's in the `pending-acceptance` state must be accepted by the owner of the peer VPC to be activated. You cannot accept a VPC peering connection request that you've sent to another AWS account. If you are creating a VPC peering connection in the same AWS account, you must both create and accept the request yourself.

> **Important**
> Do not accept VPC peering connections from AWS accounts that you do not know. A malicious user may have sent you a VPC peering connection request to gain unauthorized network access to your VPC. This is known as peer phishing. You can safely reject unwanted VPC peering connection requests without any risk of the requester gaining access to any information about your AWS account or your VPC. For more information about rejecting VPC peering requests, see Rejecting a VPC Peering Connection (p. 157). You can also ignore the request and let it expire; by default, the request expires after 7 days.

To accept a VPC peering connection

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Peering Connections**.
3. To view all VPC peering connections that are pending your acceptance, select **Requests pending my approval** from the filter list.
4. Select the VPC peering connection, and click **Accept request**.
5. In the confirmation dialog box, click **Yes, Accept**. A second confirmation dialog displays; click **Modify my route tables now** to go directly to the route tables page, or click **Close** to do this later.

Now that your VPC peering connection is active, you must add an entry to your VPC's route table to enable traffic to be directed to the peer VPC. For more information, see Updating Route Tables for Your VPC Peering Connection (p. 157).

Rejecting a VPC Peering Connection

You can reject any VPC peering connection request that you've received that's in the `pending-acceptance` state. You should only accept VPC peering connections from AWS accounts that you know and trust; you can reject any unwanted requests.

To reject a VPC peering connection

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Peering Connections**.
3. To view all VPC peering connections that are pending your acceptance, select **Requests pending my approval** from the filter list.
4. Select the VPC peering connection, and click **Reject request**.
5. In the confirmation dialog box, click **Yes, Reject**.

Updating Route Tables for Your VPC Peering Connection

To send traffic between instances in peered VPCs using private IP addresses, you must add a route to a route table that's associated with your VPC. The route points to the CIDR block (or portion of the CIDR block) of the other VPC in the VPC peering connection.

Similarly, the owner of the other VPC in the peering connection must add a route to one of their VPC's route tables to direct traffic back to your VPC. For more examples of supported route table configurations for VPC peering connections, see the *Amazon VPC Peering Guide*.

You can add a route for a VPC peering connection that's in the `pending-acceptance` state; however, the route will have a state of `blackhole` and have no effect until the VPC peering connection is in the `active` state.

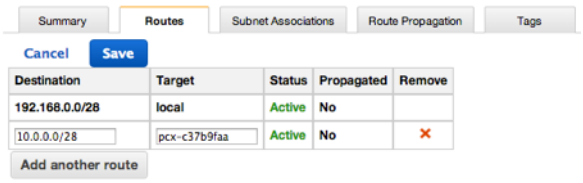
For more information about route tables, see Route Tables (p. 123).

Warning
If you have a VPC peered with multiple VPCs that have overlapping or matching CIDR blocks, ensure that your route tables are configured to avoid sending response traffic from your VPC to the incorrect VPC. AWS currently does not support unicast reverse path forwarding in VPC peering connections that checks the source IP of packets and routes reply packets back to the source. For more information, see Routing for Response Traffic in the *Amazon VPC Peering Guide*.

To add a route for a VPC peering connection using the console

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Route Tables**.
3. Select the route table that's associated with the subnet in which your instance resides.

   Note
   If you do not have a route table associated with that subnet, select the main route table for the VPC, as the subnet then uses this route table by default.
4. Click the **Routes** tab, and then click **Edit**.
5. Click **Add Route**.
6. In the **Destination** field, enter the IP address range to which the network traffic in the VPC peering connection must be directed. You can specify the entire CIDR block of the peer VPC, a specific range, or an individual IP address, such as the IP address of the instance with which to communicate. For example, if the CIDR block of the peer VPC is `10.0.0.0/16`, you can specify a portion `10.0.0.0/28`, or a specific IP address `10.0.0.7/32`.
7. Select the VPC peering connection from the **Target** list, and then click **Save**.



Describing Your VPC Peering Connections

You can view all your VPC peering connections in the VPC console. By default, the console displays all VPC peering connections in different states, including those that may have been recently deleted or rejected. For more information about the lifecycle of a VPC peering connection, see VPC Peering Connection Lifecycle (p. 154).

To view your VPC peering connections

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Peering Connections**.

3. All your VPC peering connections are listed. Use the filter lists and search field to narrow your results. For example, to view VPC peering connection requests that you've sent that are waiting for approval, select **Outstanding requests I've sent** from the filter list.

Deleting a VPC Peering Connection

Either owner of a VPC in a peering connection can delete the VPC peering connection at any time. You can also delete a VPC peering connection that you've requested that is still in the `pending-acceptance` state.

> **Note**
> Deleting a VPC in the VPC console that's part of an active VPC peering connection deletes the VPC peering connection. If you have requested a VPC peering connection with a VPC in another account, and you delete your VPC before the other party has accepted the request, the VPC peering connection is deleted. You cannot delete a VPC for which you have a `pending-acceptance` request from a VPC in another account. You must first reject the VPC peering connection request.

To delete a VPC peering connection

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Peering Connections**.
3. Select the VPC peering connection, and click **Delete**.
4. In the confirmation dialog box, click **Yes, Delete**.

API and CLI Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Create a VPC peering connection

- create-vpc-peering-connection (AWS CLI)
- ec2-create-vpc-peering-connection (Amazon EC2 CLI)
- New-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- CreateVpcPeeringConnection

Accept a VPC peering connection

- accept-vpc-peering-connection (AWS CLI)
- ec2-accept-vpc-peering-connection (Amazon EC2 CLI)
- Approve-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- AcceptVpcPeeringConnection

Describe VPC peering connections

- describe-vpc-peering-connections (AWS CLI)
- ec2-describe-vpc-peering-connections (Amazon EC2 CLI)
- Get-EC2VpcPeeringConnections (AWS Tools for Windows PowerShell)
- DescribeVpcPeeringConnections

Reject VPC peering connections

- reject-vpc-peering-connection (AWS CLI)
- ec2-reject-vpc-peering-connection (Amazon EC2 CLI)
- Deny-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- RejectVpcPeeringConnection

Delete a VPC peering connection

- delete-vpc-peering-connection (AWS CLI)
- ec2-delete-vpc-peering-connection (Amazon EC2 CLI)
- Remove-EC2VpcPeeringConnection (AWS Tools for Windows PowerShell)
- DeleteVpcPeeringConnection

Add a route to a route table

- create-route (AWS CLI)
- ec2-create-route (Amazon EC2 CLI)
- New-EC2Route (AWS Tools for Windows PowerShell)
- CreateRoute

Replace a route in a route table

- replace-route (AWS CLI)
- ec2-replace-route (Amazon EC2 CLI)
- Set-EC2Route (AWS Tools for Windows PowerShell)
- ReplaceRoute

Controlling Access to VPC Peering Connections

By default, IAM users cannot create or modify VPC peering connections. You can create an IAM policy that grants users permission to create or modify VPC peering connections, and you can control which resources users have access to during those requests. For more information about IAM policies for Amazon EC2, see IAM Policies for Amazon EC2 in the *Amazon EC2 User Guide for Linux Instances*.

For example policies for working with VPC peering connections, see Controlling Access to Amazon VPC Resources (p. 87).

VPC Endpoints

A VPC endpoint enables you to create a private connection between your VPC and another AWS service without requiring access over the Internet, through a NAT instance, a VPN connection, or AWS Direct Connect. Endpoints are virtual devices. They are horizontally scaled, redundant, and highly available VPC components that allow communication between instances in your VPC and AWS services without imposing availability risks or bandwidth constraints on your network traffic.

> **Important**
> Currently, we support endpoints for connections with Amazon S3 within the same region only. We'll add support for other AWS services later.

An endpoint enables instances in your VPC to use their private IP addresses to communicate with resources in other services. Your instances do not require public IP addresses, and you do not need an Internet gateway, a NAT instance, or a virtual private gateway in your VPC. You use endpoint policies to control access to resources in other services. Traffic between your VPC and the AWS service does not leave the Amazon network.

There is no additional charge for using endpoints. Standard charges for data transfer and resource usage apply. For more information about pricing, see Amazon EC2 Pricing.

Topics

- Endpoint Basics (p. 161)
- Controlling the Use of Endpoints (p. 165)
- Controlling Access to Services (p. 165)
- Working with Endpoints (p. 168)
- API and CLI Overview (p. 170)

Endpoint Basics

To create an endpoint, specify the VPC and the service to which you're connecting. A service is identified by a *prefix list*, or the name and ID of a service for a region. A prefix list ID uses the form `pl-xxxxxxx` and a prefix list name uses the form `com.amazonaws.<region>.<service>`. You use the prefix list name (service name) to create an endpoint.

You can attach an endpoint policy to your endpoint that allows access to some or all of the service to which you're connecting. For more information, see Using Endpoint Policies (p. 166). To control the routing of traffic between your VPC and the other service, you can specify one or more route tables that are used by the VPC to reach the endpoint. Subnets that use these route tables have access to the endpoint, and traffic from instances in these subnets to the service is then routed through the endpoint.

You can also use the VPC wizard on the dashboard of the Amazon VPC console to create a new VPC, and request an endpoint for the service that you specify.

After you've created an endpoint, you can modify the policy that's attached to your endpoint, and add or remove the route tables that are used by the endpoint.

You can create multiple endpoints in a single VPC, for example, to multiple services. You can also create multiple endpoints for a single service, and you can use different route tables to enforce different access policies from different subnets to the same service.

Topics

- Routing for Endpoints (p. 161)
- Endpoints for Amazon S3 (p. 163)
- Endpoint Limitations (p. 165)

Routing for Endpoints

When you create or modify an endpoint, you specify the VPC route tables that must be used to access the service via the endpoint. A route is automatically added to each of the route tables with a destination that specifies the prefix list ID of the service (`pl-xxxxxxxx`), and a target with the endpoint ID (`vpce-xxxxxxxx`). The prefix list ID logically represents the range of public IP addresses used by the service. All instances in subnets associated with the specified route tables automatically use the endpoint to access the service; subnets that are not associated with the specified route tables do not use the endpoint to access the service. This enables you to keep resources in other subnets separate from your endpoint.

We use the most specific route that matches the traffic to determine how to route the traffic (longest prefix match). If you have an existing route in your route table for all Internet traffic (`0.0.0.0/0`) that points to an Internet gateway, the endpoint route takes precedence for all traffic destined for the service, because the IP address range for the service is more specific than `0.0.0.0/0`. All other Internet traffic goes to your Internet gateway, including traffic that's destined for the service in other regions. However, if you have existing, more specific routes to IP address ranges that point to an Internet gateway or a NAT instance, those routes take precedence. If you have existing routes destined for an IP address range that is identical to the IP address range used by the service, then your routes take precedence.

To view the current IP address range for a service, you can use the describe-prefix-lists command or the ec2-describe-prefix-lists command.

> **Note**
> The range of public IP addresses for a service may change from time to time. Consider the implications before you make routing or other decisions based on the current IP address range for a service.

You can have multiple endpoint routes to different services in a route table, and you can have multiple endpoint routes to the same service in different route tables, but you cannot have multiple endpoints to the same service in a single route table. For example, if you have two endpoints to Amazon S3 in your VPC, you cannot use the same route table for both endpoints.

You cannot explicitly add, modify, or delete an endpoint route in your route table by using the route table APIs, or by using the Route Tables page in the VPC console. You can only add an endpoint route by associating a route table with an endpoint. The endpoint route is automatically deleted when you remove the route table association from the endpoint (by modifying the endpoint), or when you delete your endpoint.

To change the route tables that are associated with your endpoint, you can modify the endpoint. For more information, see Modifying an Endpoint (p. 169).

Example: An Endpoint Route in a Route Table

In this scenario, you have an existing route in your route table for all Internet traffic (`0.0.0.0/0`) that points to an Internet gateway. Any traffic from the subnet that's destined for another AWS service uses the Internet gateway.

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	igw-1a2b3c4d

You create an endpoint to Amazon S3, and associate your route table with the endpoint. An endpoint route is automatically added to the route table, with a destination of `pl-1a2b3c4d` (assume this represents Amazon S3). Now, any traffic from the subnet that's destined for Amazon S3 in the same region goes to the endpoint, and does not go to the Internet gateway. All other Internet traffic goes to your Internet gateway, including traffic that's destined for other services, and destined for Amazon S3 in other regions.

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	igw-1a2b3c4d
pl-1a2b3c4d	vpce-11bb22cc

Example: Adjusting Your Route Tables for Endpoints

In this scenario, you have configured your route table to enable instances in your subnet to communicate with Amazon S3 buckets through an Internet gateway. You've added a route with `54.123.165.0/24` as a destination (assume this is an IP address range currently within Amazon S3), and the Internet gateway as the target. You then create an endpoint, and associate this route table with the endpoint. An endpoint route is automatically added to the route table. You then use the describe-prefix-lists command to view the IP address range for Amazon S3. The range is `54.123.160.0/19`, which is less specific than the range that's pointing to your Internet gateway. This means that any traffic destined for the `54.123.165.0/24` IP address range continues to use the Internet gateway, and does not use the endpoint (for as long as this remains the public IP address range for Amazon S3).

Destination	Target
10.0.0.0/16	Local
54.123.165.0/24	igw-1a2b3c4d
pl-1a2b3c4d	vpce-11bb22cc

To ensure that all traffic destined for Amazon S3 in the same region is routed via the endpoint, you must adjust the routes in your route table. To do this, you can delete the route to the Internet gateway. Now, all traffic to Amazon S3 in the same region uses the endpoint, and the subnet that's associated with your route table is a private subnet.

Destination	Target
10.0.0.0/16	Local
pl-1a2b3c4d	vpce-11bb22cc

Endpoints for Amazon S3

If you've already set up access to your Amazon S3 resources from your VPC, you can continue to use Amazon S3 DNS names to access those resources after you've set up an endpoint. However, take note of the following:

- Your endpoint has a policy that controls the use of the endpoint to access Amazon S3 resources. The default policy allows access by any user or service within the VPC, using credentials from any AWS account, to any Amazon S3 resource; including Amazon S3 resources for an AWS account other than the account with which the VPC is associated. For more information, see Controlling Access to Services (p. 165).
- The source IP addresses from instances in your affected subnets as received by Amazon S3 will change from public IP addresses to the private IP addresses from your VPC. An endpoint switches network routes, and disconnects open TCP connections. Your tasks will be interrupted during the changeover, and any previous connections using public IP addresses will not be resumed. We recommend that you do not have any critical tasks running when you create or modify an endpoint; or that you test to ensure that your software can automatically reconnect to Amazon S3 after the connection break.
- If you've used IP address conditions in your bucket policies to control access to your S3 buckets (using the `aws:SourceIp` condition), access to your buckets is denied when the source IP addresses change to private IP addresses. You cannot use the `aws:SourceIp` condition in your bucket policies for private IP addresses in your VPC — that condition is ignored for any requests to Amazon S3 via the endpoint. Instead, adjust your bucket policy to limit access to a specific VPC or a specific endpoint. For more information, see Using Amazon S3 Bucket Policies (p. 166).
- You must enable DNS resolution in your VPC for endpoints to work. For more information, see Using DNS with Your VPC (p. 149).

- Endpoints currently do not support cross-region requests—ensure that you create your endpoint in the same region as your bucket. You can find the location of your bucket by using the Amazon S3 console, or by using the get-bucket-location command. Use a region-specific Amazon S3 endpoint to access your bucket; for example, `mybucket.s3-us-west-2.amazonaws.com`. For more information about region-specific endpoints for Amazon S3, see Amazon Simple Storage Service (S3) in *Amazon Web Services General Reference*. If you use the AWS CLI to make requests to Amazon S3, set your default region to the same region as your bucket, or use the `--region` parameter in your requests.

  Note
  Treat Amazon S3's US Standard region as mapped to the us-east-1 region.

If you use other AWS services in your VPC, such as Amazon Elastic MapReduce, they may use S3 buckets for certain tasks. Ensure that your endpoint policy allows full access to Amazon S3 (the default policy), or that it allows access to the specific buckets that are used by these services. Alternatively, only create an endpoint in a subnet that is not used by any of these services, to allow the services to continue accessing S3 buckets using public IP addresses.

The following table lists AWS services that may be affected by an endpoint, and any specific information for each service.

AWS service	Note
AWS CloudFormation	If you have resources in your VPC that must respond to a wait condition or custom resource request, your endpoint policy must allow at least access to the specific buckets that are used by these resources. For more information, see AWS CloudFormation and VPC Endpoints.
AWS CodeDeploy	Your endpoint policy must allow full access to Amazon S3, or allow access to any S3 buckets that you've created for your AWS CodeDeploy deployments.
Elastic Beanstalk	Your endpoint policy must allow at least access to any S3 buckets used for Elastic Beanstalk applications. For more information, see Using Elastic Beanstalk with Amazon S3 in the *AWS Elastic Beanstalk Developer Guide*.
Amazon Elastic MapReduce	Your subnet's route table must include a route to an Internet gateway, and your endpoint policy must allow full access to Amazon S3. For more information, see VPC Errors in the *Amazon Elastic MapReduce Developer Guide*.
AWS OpsWorks	Your endpoint policy must allow at least access to specific buckets that are used by AWS OpsWorks. For more information, see Running a Stack in a VPC in the *AWS OpsWorks User Guide*.
Amazon Redshift	Your endpoint policy must allow at least access to any S3 buckets used for loading data into your tables, or for creating audit log files. For more information, see Loading Data in the *Amazon Redshift Database Developer Guide*, and Database Audit Logging in the *Amazon Redshift Cluster Management Guide*.

AWS service	Note
Amazon WorkDocs	If you use an Amazon WorkDocs client in Amazon WorkSpaces or an EC2 instance, your endpoint policy must allow full access to Amazon S3.
Amazon WorkSpaces	Amazon WorkSpaces does not directly depend on Amazon S3; however, if you provide Amazon WorkSpaces users with Internet access, then take note that web sites, HTML emails, and Internet services from other companies may depend on Amazon S3. Ensure that your endpoint policy allows full access to Amazon S3 to allow these services to continue to work correctly.

Traffic between your VPC and S3 buckets does not leave the Amazon network.

Endpoint Limitations

To use endpoints, you need to be aware of the current limitations:

- You cannot use prefix list IDs to create an outbound rule in a network ACL that allows or denies outbound traffic to the service specified in an endpoint. Instead, you can use a prefix list ID in an outbound security group rule. For more information, see Security Groups (p. 168).
- You cannot create an endpoint between a VPC and an AWS service in a different region.
- You cannot tag an endpoint.
- You cannot transfer an endpoint from one VPC to another, or from one service to another.
- Endpoint connections cannot be extended out of a VPC. Resources on the other side of a VPN connection, a VPC peering connection, an AWS Direct Connect connection, or a ClassicLink connection in your VPC cannot use the endpoint to communicate with resources in the endpoint service.
- When using Amazon S3 endpoints, you cannot use a bucket policy or an IAM policy to allow access from a VPC CIDR range (the private IP address range). VPC CIDR blocks can be overlapping or identical, which may lead to unexpected results. Instead, you can use a bucket policy to restrict access to a specific endpoint or to a specific VPC, and you can use your route tables to control which instances can access resources in Amazon S3 via the endpoint.

Controlling the Use of Endpoints

By default, IAM users do not have permission to work with endpoints. You can create an IAM user policy that grants users permission to create, modify, describe, and delete endpoints. We currently do not support resource-level permissions for any of the `ec2:*VpcEndpoint*` API actions, or for the `ec2:DescribePrefixLists` action — you cannot create an IAM policy that grants users permission to use a specific endpoint or prefix list. For more information, see the following example: 8. Creating and managing VPC endpoints (p. 97).

Controlling Access to Services

When you create an endpoint, you attach an endpoint policy to it that controls access to the service to which you are connecting. If you're using an endpoint to Amazon S3, you can also use Amazon S3 bucket policies to control access to buckets from specific endpoints, or specific VPCs.

Endpoint policies and Amazon S3 bucket policies must be written in JSON format.

Topics

- Using Endpoint Policies (p. 166)
- Using Amazon S3 Bucket Policies (p. 166)
- Security Groups (p. 168)

Using Endpoint Policies

A VPC endpoint policy is an IAM resource policy that you attach to an endpoint when you create or modify the endpoint. If you do not attach a policy when you create an endpoint, we attach a default policy for you that allows full access to the service. An endpoint policy does not override or replace IAM user policies or S3 bucket policies. It is a separate policy for controlling access from the endpoint to the specified service. However, all types of policies — IAM user policies, endpoint policies, S3 policies, and Amazon S3 ACL policies (if any) — must grant the necessary permissions for access to Amazon S3 to succeed.

You cannot attach more than one policy to an endpoint; however, you can modify the policy at any time. Note that if you do modify a policy, it can take a few minutes for the changes to take effect. For more information, see Modifying an Endpoint (p. 169). For more information about writing policies, see Overview of AWS IAM Policies in *IAM User Guide*.

Your endpoint policy can be like any IAM policy; however, take note of the following:

- Only the parts of the policy that relate to the specified service will work. You cannot use an endpoint policy to allow resources in your VPC to perform other actions; for example, if you add EC2 actions to an endpoint policy for an endpoint to Amazon S3, they will have no effect.
- Your policy must contain a `Principal` element. For more information, see Principal in *IAM User Guide*.

Example: Restricting Access to a Specific Bucket

You can create a policy that restricts access to specific S3 buckets only. This is useful if you have other AWS services in your VPC that use S3 buckets. The following is an example of a policy that restricts access to `my_secure_bucket` only.

```
{
  "Statement": [
    {
      "Sid": "Access-to-specific-bucket-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::my_secure_bucket",
                   "arn:aws:s3:::my_secure_bucket/*"]
    }
  ]
}
```

Using Amazon S3 Bucket Policies

You can use bucket policies to control access to buckets from specific endpoints, or specific VPCs.

For more information about bucket policies for Amazon S3, see Using Bucket Policies and User Policies in *Amazon Simple Storage Service Developer Guide*.

Example: Restricting Access to a Specific Endpoint

The following is an example of an S3 bucket policy that allows access to a specific bucket, `my_secure_bucket`, from endpoint `vpce-1a2b3c4d` only. The policy uses the `aws:sourceVpce` condition to restrict access to the specified endpoint. The `aws:sourceVpce` condition does not require an ARN for the VPC endpoint resource, only the endpoint ID.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909152",
  "Statement": [
    {
      "Sid": "Access-to-specific-VPCE-only",
      "Principal": "*",
      "Action": "s3:*",
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::my_secure_bucket",
                   "arn:aws:s3:::my_secure_bucket/*"],
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpce": "vpce-1a2b3c4d"
        }
      }
    }
  ]
}
```

Example: Restricting Access to a Specific VPC

You can create a bucket policy that restricts access to a specific VPC by using the `aws:sourceVpc` condition. This is useful if you have multiple endpoints configured in the same VPC, and you want to manage access to your S3 buckets for all of your endpoints. The following is an example of a policy that allows VPC `vpc-111bbb22` to access `my_secure_bucket` and its objects. The `aws:sourceVpc` condition does not require an ARN for the VPC resource, only the VPC ID.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909152",
  "Statement": [
    {
      "Sid": "Access-to-specific-VPC-only",
      "Principal": "*",
      "Action": "s3:*",
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::my_secure_bucket",
                   "arn:aws:s3:::my_secure_bucket/*"],
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpc": "vpc-111bbb22"
        }
      }
    }
  ]
}
```

Security Groups

By default, Amazon VPC security groups allow all outbound traffic, unless you've specifically restricted outbound access. If your security group's outbound rules are restricted, you must add a rule that allows outbound traffic from your VPC to the service that's specified in your endpoint. To do this, you can use the service's prefix list ID as the destination in the outbound rule. For more information, see Modifying Your Security Group (p. 169).

Working with Endpoints

You can use the Amazon VPC console to create and manage endpoints.

Topics

- Creating an Endpoint (p. 168)
- Modifying Your Security Group (p. 169)
- Modifying an Endpoint (p. 169)
- Describing Your Endpoints (p. 170)
- Deleting an Endpoint (p. 170)

Creating an Endpoint

To create an endpoint, you must specify the VPC in which you want to create the endpoint, and the service to which you want to establish the connection. You can also attach a policy to the endpoint, and specify the route tables that will be used by the endpoint.

To create an endpoint

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Endpoints**.
3. Choose **Create Endpoint**.
4. In the first step of the wizard, complete the following information, and then choose **Next Step**.
   - Select a VPC in which to create the endpoint, and the service to which you want to connect. (Currently, only the Amazon S3 service is available.)
   - Choose the type of policy. You can leave the default option, **Full Access**, to allow full access to the service. Alternatively, you can select **Custom**, and then use the AWS Policy Generator to create a custom policy, or type your own policy in the policy window.
5. In the second step of the wizard, select the route tables that will be used by the endpoint. The wizard automatically adds a route to those tables that points traffic destined for the service to the endpoint. When you are done, choose **Create Endpoint**.

You can use the VPC wizard to create a new VPC and to create an endpoint at the same time. Instead of specifying the route tables that are used by the endpoint, you specify the subnets that will have access to the endpoint. The wizard adds an endpoint route to the route tables associated with those subnets.

To create a VPC and endpoint using the VPC wizard

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. On the Amazon VPC dashboard, choose **Start VPC Wizard**.
3. Select a VPC configuration that suit your needs, and then choose **Select**. For more information about the types of configurations, see VPC Wizard Scenarios for Amazon VPC (p. 17).

4. On the second page of the wizard, fill in the VPC settings as required. In the **Add endpoints for S3 to your subnets** section, complete the following information:
   - Select the subnets that will have access to the endpoint from the **Subnet** list. The route tables associated with the subnets will include an endpoint route.
   - Select the type of policy from the **Policy** list. You can leave the default option, **Full Access**, to allow full access to the service. Alternatively, choose **Custom**, and then use the AWS Policy Generator to create a custom policy, or type your own policy in the policy window.
5. If applicable, complete the rest of the steps in the wizard, and then click **Create VPC**.

Modifying Your Security Group

If your VPC security group restricts outbound traffic, you must add a rule to allow traffic destined for the AWS service to leave your instance.

To add an outbound rule for an endpoint

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Security Groups**.
3. Select your VPC security group, choose the **Outbound Rules** tab, and then choose **Edit**.
4. Select the type of traffic from the **Type** list, and enter the port range, if required. For example, if you use your instance to retrieve objects from Amazon S3, choose **HTTPS** from the **Type** list.
5. The **Destination** list displays the prefix list IDs and names for the available AWS services. Choose the prefix list ID for the endpoint service, or type it in.

   Note
   For Amazon S3, the prefix list name is `com.amazonaws.<region>.s3`; for example, `com.amazonaws.us-east-1.s3`.
6. Choose **Save**.

For more information about security groups, see Security Groups for Your VPC (p. 64).

Modifying an Endpoint

You can modify your endpoint by changing or removing its policy, and adding or removing the route tables that are used by the endpoint.

To change the policy associated with an endpoint

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Endpoints**.
3. Select your endpoint, choose **Actions**, and then choose **Edit Policy**.
4. In the dialog box, you can choose **Full Access** to allow full access. Alternatively, choose **Custom**, and then use the AWS Policy Generator to create a custom policy, or type your own policy in the policy window. When you're done, choose **Save Policy**.

   Note
   It can take a few minutes for policy changes to take effect.

To add or remove route tables used by an endpoint

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.

2. In the navigation pane, choose **Endpoints**.
3. Select your VPC endpoint, choose **Actions**, and then choose **Choose Route Tables**.
4. In the dialog box, select or deselect the required route tables, and then choose **Save**.

Describing Your Endpoints

You can use the Amazon VPC console to view your endpoints, and to view information about each one.

To view information about an endpoint

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Endpoints**.
3. Select your endpoint.
4. You can view information about the endpoint on the **Summary** tab; for example, you can get the prefix list name for the service in the **Service** field.

   On the **Route Tables** tab, you can view information about the route tables that are used by the endpoint. On the **Policy** tab, you can view the IAM policy that's attached to your endpoint.

   Note
   The **Policy** tab only displays the endpoint policy. It does not display any information about IAM policies for IAM users that have permission to work with endpoints. It also does not display service-specific policies; for example, S3 bucket policies.

Deleting an Endpoint

If you no longer require an endpoint, you can delete it. Deleting an endpoint also deletes the endpoint routes in the route tables that were used by the endpoint, but doesn't affect any security groups associated with the VPC in which the endpoint resides.

To delete an endpoint

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, choose **Endpoints**.
3. Select your endpoint, choose **Actions**, and then choose **Delete Endpoint**.
4. In the confirmation dialog box, choose **Yes, Delete**.

API and CLI Overview

You can perform the tasks described on this page using a command line tool, or the Amazon EC2 Query API.

Create a VPC endpoint

- create-vpc-endpoint (AWS CLI)
- ec2-create-vpc-endpoint (Amazon EC2 CLI)
- New-EC2VpcEndpoint (AWS Tools for Windows PowerShell)
- CreateVpcEndpoint (Amazon EC2 Query API)

Get the prefix list name, ID, and IP address range for an AWS service

- describe-prefix-lists (AWS CLI)
- ec2-describe-prefix-lists (Amazon EC2 CLI)
- Get-EC2PrefixList (AWS Tools for Windows PowerShell)
- DescribePrefixLists (Amazon EC2 Query API)

Modify a VPC endpoint

- modify-vpc-endpoint (AWS CLI)
- ec2-modify-vpc-endpoint (Amazon EC2 CLI)
- Edit-EC2VpcEndpoint (AWS Tools for Windows PowerShell)
- ModifyVpcEndpoint (Amazon EC2 Query API)

Describe your VPC endpoints

- describe-vpc-endpoints (AWS CLI)
- ec2-describe-vpc-endpoints (Amazon EC2 CLI)
- Get-EC2VpcEndpoint (AWS Tools for Windows PowerShell)
- DescribeVpcEndpoints (Amazon EC2 Query API)

Get a list of available AWS services for creating a VPC endpoint

- describe-vpc-endpoint-services (AWS CLI)
- ec2-describe-vpc-endpoint-services (Amazon EC2 CLI)
- Get-EC2VpcEndpointService (AWS Tools for Windows PowerShell)
- DescribeVpcEndpointServices (Amazon EC2 Query API)

Delete a VPC endpoint

- delete-vpc-endpoints (AWS CLI)
- ec2-delete-vpc-endpoints (Amazon EC2 CLI)
- Remove-EC2VpcEndpoint (AWS Tools for Windows PowerShell)
- DeleteVpcEndpoints (Amazon EC2 Query API)

Adding a Hardware Virtual Private Gateway to Your VPC

By default, instances that you launch into a virtual private cloud (VPC) can't communicate with your own network. You can enable access to your network from your VPC by attaching a virtual private gateway to the VPC, creating a custom route table, and updating your security group rules.

You can complete this process manually, as described on this page, or let the VPC creation wizard take care of many of these steps for you. For more information about using the VPC creation wizard to set up the virtual private gateway, see Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access (p. 32) or Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access (p. 41).

Although the term *VPN connection* is a general term, in the Amazon VPC documentation, a VPN connection refers to the connection between your VPC and your own network.

Topics

- Components of Your VPN (p. 173)
- VPN Configuration Examples (p. 173)
- VPN Routing Options (p. 174)
- What You Need for a VPN Connection (p. 175)
- Configuring Two VPN Tunnels for Your VPN Connection (p. 176)
- Using Redundant VPN Connections to Provide Failover (p. 177)
- Setting Up the VPN Connection (p. 177)
- Testing the End-to-End Connectivity of Your Instance (p. 180)
- Replacing Compromised Credentials (p. 181)
- Editing Static Routes for a VPN Connection (p. 181)
- Deleting a VPN Connection (p. 182)
- API and CLI Overview (p. 182)

For information about how you're charged for using a VPN connection with your VPC, see the Amazon VPC product page.

Components of Your VPN

A VPN connection consists of the following components.

Virtual Private Gateway

A *virtual private gateway* is the VPN concentrator on the Amazon side of the VPN connection.

For information about how many virtual private gateways you can have per region, as well as the limits for other components within your VPC, see Amazon VPC Limits (p. 194).

Customer Gateway

A *customer gateway* is a physical device or software application on your side of the VPN connection. When you create a VPN connection, the VPN tunnel comes up when traffic is generated from your side of the VPN connection. The virtual private gateway is not the initiator; your customer gateway must initiate the tunnels. If your VPN connection experiences a period of idle time (usually 10 seconds, depending on your configuration), the tunnel may go down. To prevent this, you can use a network monitoring tool to generate keepalive pings; for example, by using IP SLA.

For more information about customer gateways, see Your Customer Gateway in the *Amazon VPC Network Administrator Guide*.

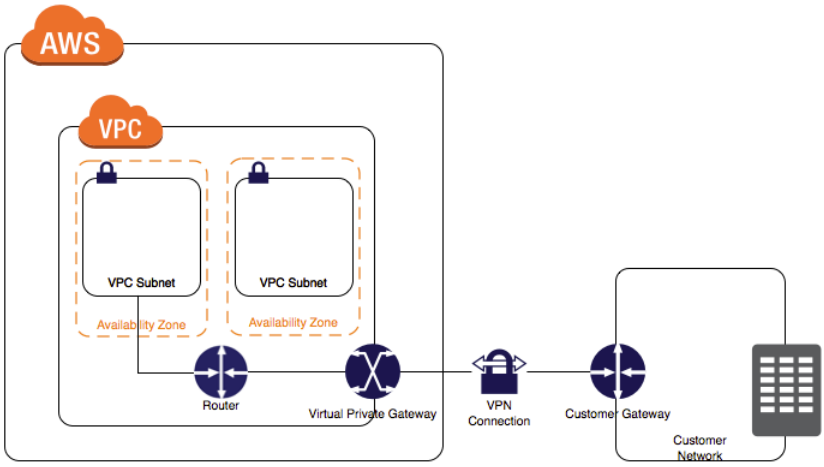
For a list of customer gateways that we have tested with Amazon VPC, see Amazon Virtual Private Cloud FAQs.

VPN Configuration Examples

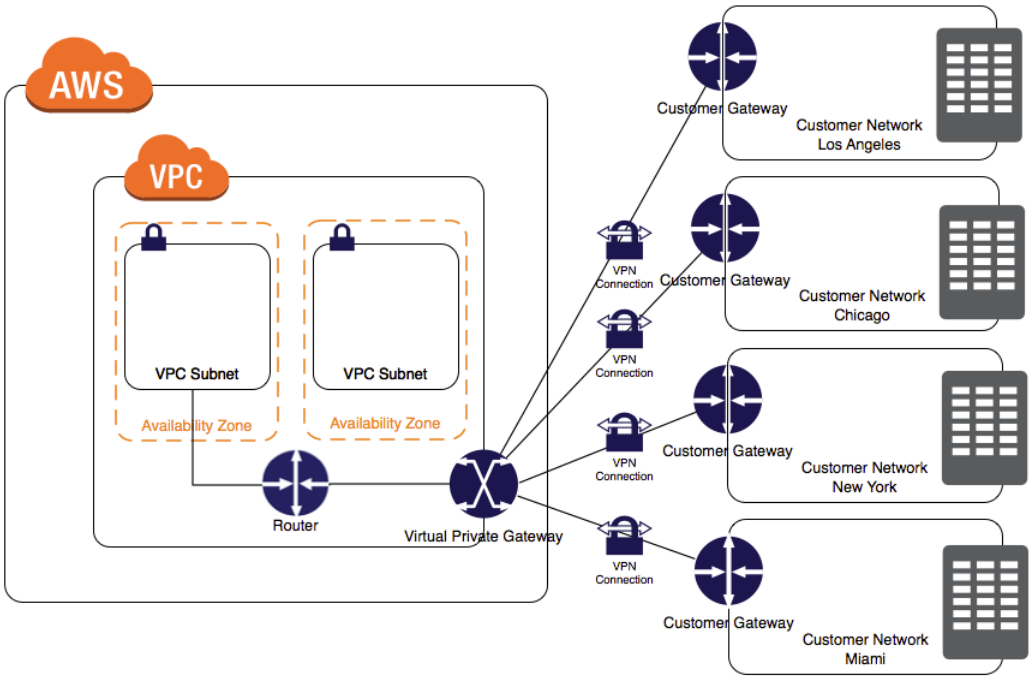
The following diagrams illustrate single and multiple VPN connections. The VPC has an attached virtual private gateway, and your network includes a customer gateway, which you must configure to enable the VPN connection. You set up the routing so that any traffic from the VPC bound for your network is routed to the virtual private gateway.

When you create multiple VPN connections to a single VPC, you can configure a second customer gateway to create a redundant connection to the same external location. You can also use it to create VPN connections to multiple geographic locations.

Single VPN Connection



Multiple VPN connections



VPN Routing Options

When you create a VPN connection, you must specify the type of routing that you plan to use. The type of routing that you select can depend on the make and model of your VPN devices. If your VPN device supports Border Gateway Protocol (BGP), specify dynamic routing when you configure your VPN connection. If your device does not support BGP, specify static routing. For a list of static and dynamic routing devices that have been tested with Amazon VPC, see the Amazon Virtual Private Cloud FAQs.

When you use a BGP device, you don't need to specify static routes to the VPN connection because the device uses BGP to advertise its routes to the virtual private gateway. If you use a device that doesn't support BGP, you must select static routing and enter the routes (IP prefixes) for your network that should be communicated to the virtual private gateway. Only IP prefixes that are known to the virtual private gateway, whether through BGP advertisement or static route entry, can receive traffic from your VPC.

We recommend that you use BGP-capable devices, when available, because the BGP protocol offers robust liveness detection checks that can assist failover to the second VPN tunnel if the first tunnel goes down. Devices that don't support BGP may also perform health checks to assist failover to the second tunnel when needed.

What You Need for a VPN Connection

To use Amazon VPC with a VPN connection, you or your network administrator must designate a physical appliance as your customer gateway and configure it. We provide you with the required configuration information, including the VPN preshared key and other parameters related to setting up the VPN connection. Your network administrator typically performs this configuration. For information about the customer gateway requirements and configuration, see the Amazon VPC Network Administrator Guide.

The following table lists the information that you need to have so that we can establish your VPN connection.

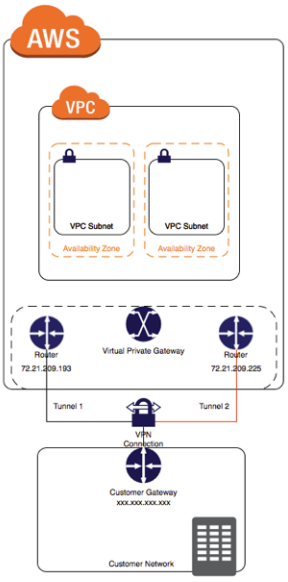
Item	How Used	Comments
The type of customer gateway (for example, Cisco ASA, Juniper J-Series, Juniper SSG, Yamaha)	Specifies how to format the returned information that you use to configure the customer gateway.	For information about the specific devices that we've tested, see What customer gateway devices are known to work with Amazon VPC? in the Amazon VPC FAQ.
Internet-routable IP address (static) of the customer gateway's external interface.	Used to create and configure your customer gateway (it's referred to as YOUR_UPLINK_ADDRESS)	The IP address value must be static and may be behind a device performing network address translation (NAT); however, NAT traversal (NAT-T) is not supported. The IP address value must be unique within the region. If the IP address is already in use by another VPN connection in any AWS account in the same region, you will get an `InvalidCustomerGateway.DuplicateIpAddress` error when you try to create the VPN connection. For more information, see Your Customer Gateway in the *Amazon VPC Network Administrator Guide*.

Item	How Used	Comments
(Optional) Border Gateway Protocol (BGP) Autonomous System Number (ASN) of the customer gateway, if you are creating a dynamically routed VPN connection.	Used to create and configure your customer gateway (referred to as YOUR_BGP_ASN). If you use the wizard in the console to set up your VPC, we automatically use 65000 as the ASN.	You can use an existing ASN assigned to your network. If you don't have one, you can use a private ASN (in the 64512–65534 range). For more information about ASNs, see the Wikipedia article. Amazon VPC supports 2-byte ASN numbers.
Internal network IP ranges that you want advertised over the VPN connection to the VPC.	Used to specify static routes.	

Configuring Two VPN Tunnels for Your VPN Connection

You use a VPN connection to connect your network to a VPC. Each VPN connection has two tunnels, with each tunnel using a unique virtual private gateway public IP address. It is important to configure both tunnels for redundancy. When one tunnel becomes unavailable (for example, down for maintenance), network traffic is automatically routed to the available tunnel for that specific VPN connection.

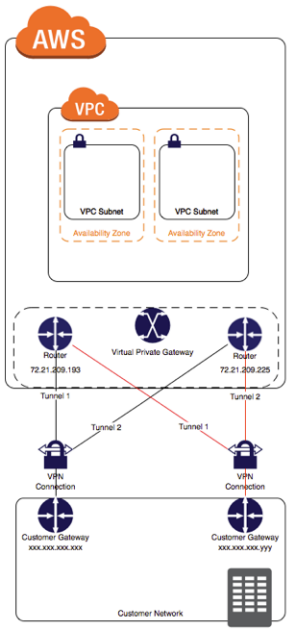
The following diagram shows the two tunnels of the VPN connection.



Using Redundant VPN Connections to Provide Failover

As described earlier, a VPN connection has two tunnels to help ensure connectivity in case one of the VPN connections becomes unavailable. To protect against a loss of connectivity in case your customer gateway becomes unavailable, you can set up a second VPN connection to your VPC by using a second customer gateway. By using redundant VPN connections and customer gateways, you can perform maintenance on one of your customer gateways while traffic continues to flow over the second customer gateway's VPN connection. To establish redundant VPN connections and customer gateways on your network, you need to set up a second VPN connection. The customer gateway IP address for the second VPN connection must be publicly accessible and can't be the same public IP address that you are using for the first VPN connection.

The following diagram shows the two tunnels of the VPN connection and two customer gateways.



Dynamically routed VPN connections use the Border Gateway Protocol (BGP) to exchange routing information between your customer gateways and the virtual private gateways. Statically routed VPN connections require you to enter static routes for the network on your side of the customer gateway. BGP-advertised and statically entered route information allow gateways on both sides to determine which tunnels are available and reroute traffic if a failure occurs. We recommend that you configure your network to use the routing information provided by BGP (if available) to select an available path. The exact configuration depends on the architecture of your network.

Setting Up the VPN Connection

Use the following procedures to manually set up the VPN connection. Alternatively, you can create the VPC and subnets and complete the first five steps in this procedure using the VPC wizard. For more information, see Implementing Scenario 3 (p. 37) or Implementing Scenario 4 (p. 44).

To set up a VPN connection, you need to complete the following steps:

- Step 1: Create a Customer Gateway (p. 178)
- Step 2: Create a Virtual Private Gateway (p. 178)
- Step 3: Enable Route Propagation in Your Route Table (p. 178)
- Step 4: Update Your Security Group to Enable Inbound SSH, RDP and ICMP Access (p. 179)
- Step 5: Create a VPN Connection and Configure the Customer Gateway (p. 179)
- Step 6: Launch an Instance Into Your Subnet (p. 180)

These procedures assume that you have a VPC with one or more subnets, and that you have the required network information (see What You Need for a VPN Connection (p. 175)).

Create a Customer Gateway

To create a customer gateway

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Customer Gateways**, and then click **Create Customer Gateway**.
3. In the **Create Customer Gateway** dialog box, complete the following and then click **Yes, Create**:
   - In the **Name tag** field, optionally enter a name for your customer gateway. Doing so creates a tag with a key of `Name` and the value that you specify.
   - Select the routing type from the **Routing** list.
   - If you selected dynamic routing, enter the Border Gateway Protocol (BGP) Autonomous System Number (ASN) in the **BGP ASN** field.
   - Enter the static, Internet-routable IP address for your customer gateway device in the **IP Address** field. The address may be behind a device performing network address translation (NAT); however, NAT traversal (NAT-T) is not supported.

Create a Virtual Private Gateway

To create a virtual private gateway

1. In the navigation pane, click **Virtual Private Gateways**, and then click **Create Virtual Private Gateway**.
2. You can optionally enter a name for your virtual private gateway, and then click **Yes, Create**.
3. Select the virtual private gateway that you created, and then click **Attach to VPC**.
4. In the **Attach to VPC** dialog box, select your VPC from the list, and then click **Yes, Attach**.

Enable Route Propagation in Your Route Table

To enable instances in your VPC to reach your customer gateway, you must configure your route table to include the routes used by your VPN connection and point them to your virtual private gateway. You can enable route propagation for your route table to automatically propagate those routes to the table for you.

For static routing, the static IP prefixes that you specify for your VPN configuration are propagated to the route table after you've created the VPN connection. For dynamic routing, the BGP-advertised routes from your customer gateway are propagated to the route table when the status of the VPN connection is `UP`.

To enable route propagation

1. In the navigation pane, click **Route Tables**, and then select the route table that's associated with the subnet; by default, this is the main route table for the VPC.
2. On the **Route Propagation** tab in the details pane, click **Edit**, select the virtual private gateway that you created in the previous procedure, and then click **Save**.

Note
For static routing, if you do not enable route propagation, you must manually enter the static routes used by your VPN connection. To do this, select your route table, then on the **Routes** tab in the details pane, click **Edit**. Add the static route used by your VPN connection in the **Destination** field, select the virtual private gateway ID from the **Target** list, and then click **Save**.

Update Your Security Group to Enable Inbound SSH, RDP and ICMP Access

To add rules to your security group to enable inbound SSH, RDP and ICMP access

1. In the navigation pane, click **Security Groups**, and then select the default security group for the VPC.
2. On the **Inbound** tab in the details pane, add rules that allow inbound SSH, RDP and ICMP access from your network, and then click **Save**. For more information about adding inbound rules, see Adding and Removing Rules (p. 67).

Create a VPN Connection and Configure the Customer Gateway

To create a VPN connection and configure the customer gateway

1. In the navigation pane, click **VPN Connections**.
2. Click **Create VPN Connection**.
3. In the **Create VPN Connection** dialog box, do the following, and then click **Yes, Create**:
   - In the **Name tag** field, optionally enter a name for your VPN connection. Doing so creates a tag with a key of `Name` and the value that you specify.
   - Select the virtual private gateway that you created earlier.
   - Select the customer gateway that you created earlier.
   - Select one of the routing options based on whether your VPN router supports Border Gateway Protocol (BGP):
     - If your VPN router supports BGP, select **Dynamic (requires BGP)**.
     - If your VPN router does not support BGP, select **Static**. In the **Static IP Prefixes** field, specify each IP prefix for the private network of your VPN connection, separated by commas.
4. It may take a few minutes to create the VPN connection. When it's ready, select the connection, and then click **Download Configuration**.
5. In the **Download Configuration** dialog box, select the vendor, platform, and software that corresponds to your customer gateway device or software, and then click **Yes, Download**.

6. Give the configuration file to your network administrator, along with this guide: Amazon VPC Network Administrator Guide. After the network administrator configures the customer gateway, the VPN connection is operational.

Launch an Instance Into Your Subnet

To launch an instance into your subnet

1. Open the Amazon EC2 console.
2. On the dashboard, click **Launch Instance**.
3. On the **Choose an Amazon Machine Image (AMI)** page, choose an AMI, and then click **Select**.
4. Choose an instance type, and then click **Next: Configure Instance Details**.
5. On the **Configure Instance Details** page, select your VPC from the **Network** list, and your subnet from the **Subnet** list. Click **Next** until you reach the **Configure Security Group** page.
6. Select the **Select an existing security group** option, and then select the default group that you modified earlier. Click **Review and Launch**.
7. Review the settings that you've chosen. Make any changes that you need, and then click **Launch** to select a key pair and launch the instance.

Testing the End-to-End Connectivity of Your Instance

After you set up your VPN connection and launch an instance, you can test the connection by pinging the instance. You need to use an AMI that responds to ping requests, and you need to ensure that your instance's security group is configured to enable inbound ICMP. We recommend you use one of the Amazon Linux AMIs. If you are using instances running Windows Server, you'll need to log in to the instance and enable inbound ICMPv4 on the Windows firewall in order to ping the instance.

> **Important**
> You must configure any security group or network ACL in your VPC that filters traffic to the instance to allow inbound and outbound ICMP traffic.

You can monitor the status of your VPN connections using the Amazon VPC console or by using the Amazon EC2 API/CLI. You can view information about your VPN connections, including its state, the time since last state change, and descriptive error text.

To test end-to-end connectivity

1. After the instance is running, get its private IP address (for example, `10.0.0.4`). The Amazon EC2 console displays the address as part of the instance's details.
2. From a computer in your network that is behind the customer gateway, use the **ping** command with the instance's private IP address. A successful response is similar to the following:

```
PROMPT> ping 10.0.0.4
Pinging 10.0.0.4 with 32 bytes of data:

Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128
Reply from 10.0.0.4: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.4:
```

```
Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),

Approximate round trip times in milliseconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

You can now use SSH or RDP to connect to your instance in the VPC. For more information about how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For more information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Replacing Compromised Credentials

If you believe that the tunnel credentials for your VPN connection have been compromised, you can change the IKE preshared key. To do so, delete the VPN connection, create a new one using the same virtual private gateway, and configure the new keys on your customer gateway. You also need to confirm that the tunnel's inside and outside addresses match, because these might change when you recreate the VPN connection. While you perform the procedure, communication with your instances in the VPC stops, but the instances continue to run uninterrupted. After the network administrator implements the new configuration information, your VPN connection uses the new credentials, and the network connection to your instances in the VPC resumes.

Important
This procedure requires assistance from your network administrator group.

To change the IKE pre-shared key

1. Delete the VPN connection. For more information, see Deleting a VPN Connection (p. 182). You don't need to delete the VPC or the virtual private gateway.
2. Create a new VPN connection and download the new configuration file. For more information, see Create a VPN Connection and Configure the Customer Gateway (p. 179).

Editing Static Routes for a VPN Connection

For static routing, you can add, modify, or remove the static routes for your VPN configuration.

To add, modify, or remove a static route

1. In the navigation pane, click **VPN Connections**.
2. In the **Static Routes** tab, click **Edit**.
3. Modify your existing static IP prefixes, or click **Remove** to delete them. Click **Add Another Route** to add a new IP prefix to your configuration. When you are done, click **Save**.

Note
If you have not enabled route propagation for your route table, you must manually update the routes in your route table to reflect the updated static IP prefixes in your VPN connection. For more information, see Enable Route Propagation in Your Route Table (p. 178).

Deleting a VPN Connection

If you no longer need a VPN connection, you can delete it.

Important
If you delete your VPN connection and then create a new one, you have to download new configuration information and have your network administrator reconfigure the customer gateway.

To delete a VPN connection

1. Open the Amazon VPC console.
2. In the navigation pane, click **VPN Connections**.
3. Select the VPN connection and click **Delete**.
4. In the **Delete VPN Connection** dialog box, click **Yes, Delete**.

If you no longer require a customer gateway, you can delete it. You can't delete a customer gateway that's being used in a VPN connection.

To delete a customer gateway

1. In the navigation pane, click **Customer Gateways**.
2. Select the customer gateway to delete and click **Delete**.
3. In the **Delete Customer Gateway** dialog box, click **Yes, Delete**.

If you no longer require a virtual private gateway for your VPC, you can detach it.

To detach a virtual private gateway

1. In the navigation pane, click **Virtual Private Gateways**.
2. Select the virtual private gateway and click **Detach from VPC**.
3. In the **Detach from VPC** dialog box, click **Yes, Detach**.

If you no longer require a detached virtual private gateway, you can delete it. You can't delete a virtual private gateway that's still attached to a VPC.

To delete a virtual private gateway

1. Select the virtual private gateway to delete and click **Delete**.
2. In the **Delete Virtual Private Gateway** dialog box, click **Yes, Delete**.

API and CLI Overview

You can use the command line or an API action to set up and manage your VPN connection. For more information, including a list of available API actions, see Accessing Amazon VPC (p. 6).

Create a customer gateway

- CreateCustomerGateway (Amazon EC2 Query API)
- ec2-create-customer-gateway (Amazon EC2 CLI)
- create-customer-gateway (AWS CLI)
- New-EC2CustomerGateway (AWS Tools for Windows PowerShell)

Create a virtual private gateway

- CreateVpnGateway (Amazon EC2 Query API)
- ec2-create-vpn-gateway (Amazon EC2 CLI)
- create-vpn-gateway (AWS CLI)
- New-EC2VpnGateway (AWS Tools for Windows PowerShell)

Enable route propagation

- EnableVgwRoutePropagation (Amazon EC2 Query API)
- ec2-enable-vgw-route-propagation (Amazon EC2 CLI)
- enable-vgw-route-propagation (AWS CLI)
- Enable-EC2VgwRoutePropagation (AWS Tools for Windows PowerShell)

Update your security group

- For more information about working with security groups using a CLI, see API and CLI Overview (p. 69).

Create a VPN connection

- CreateVpnConnection (Amazon EC2 Query API)
- ec2-create-vpn-connection (Amazon EC2 CLI)
- create-vpn-connection (AWS CLI)
- New-EC2VpnConnection (AWS Tools for Windows PowerShell)

Add a static route

- CreateVpnConnectionRoute (Amazon EC2 Query API)
- ec2-create-vpn-connection-route (Amazon EC2 CLI)
- create-vpn-connection-route (AWS CLI)
- New-EC2VpnConnectionRoute (AWS Tools for Windows PowerShell)

Delete a static route

- DeleteVpnConnectionRoute (Amazon EC2 Query API)
- ec2-delete-vpn-connection-route (Amazon EC2 CLI)
- delete-vpn-connection-route (AWS CLI)
- Remove-EC2VpnConnectionRoute (AWS Tools for Windows PowerShell)

Delete a VPN connection

- DeleteVpnConnection (Amazon EC2 Query API)
- ec2-delete-vpn-connection (Amazon EC2 CLI)
- delete-vpn-connection (AWS CLI)
- Remove-EC2VpnConnection (AWS Tools for Windows PowerShell)

Delete a customer gateway

- DeleteCustomerGateway (Amazon EC2 Query API)

- ec2-delete-customer-gateway (Amazon EC2 CLI)
- delete-customer-gateway (AWS CLI)
- Remove-EC2CustomerGateway (AWS Tools for Windows PowerShell)

Detach a virtual private gateway

- DetachVpnGateway (Amazon EC2 Query API)
- ec2-detach-vpn-gateway (Amazon EC2 CLI)
- detach-vpn-gateway (AWS CLI)
- Dismount-EC2VpnGateway (AWS Tools for Windows PowerShell)

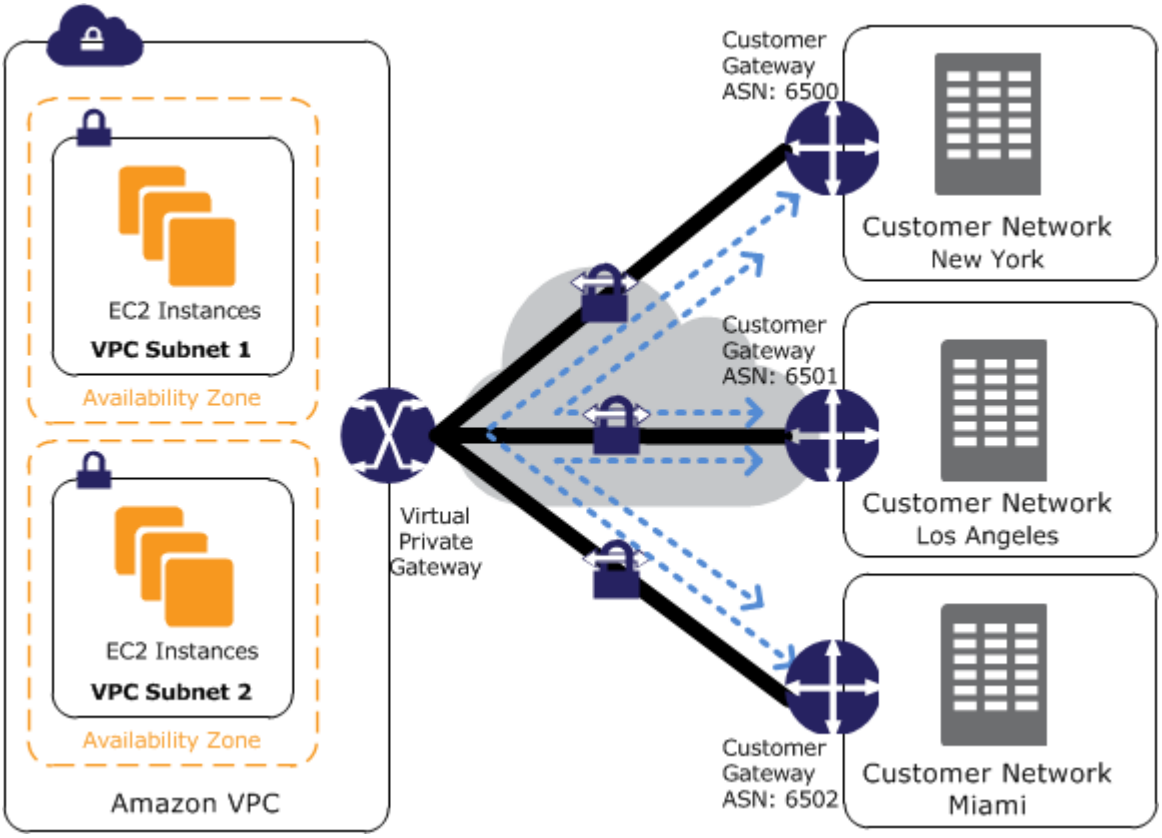
Delete a virtual private gateway

- DeleteVpnGateway (Amazon EC2 Query API)
- ec2-delete-vpn-gateway (Amazon EC2 CLI)
- delete-vpn-gateway (AWS CLI)
- Remove-EC2VpnGateway (AWS Tools for Windows PowerShell)

Providing Secure Communication Between Sites Using VPN CloudHub

If you have multiple VPN connections, you can provide secure communication between sites using the AWS VPN CloudHub. The VPN CloudHub operates on a simple hub-and-spoke model that you can use with or without a VPC. This design is suitable for customers with multiple branch offices and existing Internet connections who'd like to implement a convenient, potentially low-cost hub-and-spoke model for primary or backup connectivity between these remote offices.

The following diagram shows the VPN CloudHub architecture, with blue dashed lines indicating network traffic between remote sites being routed over their VPN connections.



To use the AWS VPN CloudHub, you must create a virtual private gateway with multiple customer gateways, each with unique Border Gateway Protocol (BGP) Autonomous System Numbers (ASNs). Customer gateways advertise the appropriate routes (BGP prefixes) over their VPN connections. These routing advertisements are received and re-advertised to each BGP peer, enabling each site to send data to and receive data from the other sites. The routes for each spoke must have unique ASNs and the sites must not have overlapping IP ranges. Each site can also send and receive data from the VPC as if they were using a standard VPN connection.

Sites that use AWS Direct Connect connections to the virtual private gateway can also be part of the AWS VPN CloudHub. For example, your corporate headquarters in New York can have an AWS Direct Connect connection to the VPC and your branch offices can use VPN connections to the VPC. The branch offices in Los Angeles and Miami can send and receive data with each other and with your corporate headquarters, all using the AWS VPN CloudHub.

To configure the AWS VPN CloudHub, you use the AWS Management Console to create multiple customer gateways, each with the unique public IP address of the gateway and a unique ASN. Next, you create a VPN connection from each customer gateway to a common virtual private gateway. Each VPN connection must advertise its specific BGP routes. This is done using the network statements in the VPN configuration files for the VPN connection. The network statements differ slightly depending on the type of router you use.

When using an AWS VPN CloudHub, you pay typical Amazon VPC VPN connection rates. You are billed the connection rate for each hour that each VPN is connected to the virtual private gateway. When you send data from one site to another using the AWS VPN CloudHub, there is no cost to send data from your site to the virtual private gateway. You only pay standard AWS data transfer rates for data that is relayed from the virtual private gateway to your endpoint. For example, if you have a site in Los Angeles and a second site in New York and both sites have a VPN connection to the virtual private gateway, you pay $.05 per hour for each VPN connection (for a total of $.10 per hour). You also pay the standard AWS data transfer rates for all data that you send from Los Angeles to New York (and vice versa) that traverses

each VPN connection; network traffic sent over the VPN connection to the virtual private gateway is free but network traffic sent over the VPN connection from the virtual private gateway to the endpoint is billed at the standard AWS data transfer rate. For more information, see VPN Connection Pricing.

Dedicated Instances

Dedicated Instances are Amazon EC2 instances that run in a virtual private cloud (VPC) on hardware that's dedicated to a single customer. Your Dedicated Instances are physically isolated at the host hardware level from your instances that aren't Dedicated Instances and from instances that belong to other AWS accounts.

This topic discusses the basics of Dedicated Instances and shows you how to implement them.

Topics

- Dedicated Instance Basics (p. 188)
- Working with Dedicated Instances (p. 189)
- API and Command Overview (p. 191)

Dedicated Instance Basics

Each instance that you launch into a VPC has a tenancy attribute. You can't change the tenancy of an instance after you launch it. This attribute has the following values.

Value	Description
`default`	Your instance runs on shared hardware.
`dedicated`	Your instance runs on single-tenant hardware.

Each VPC has a related instance tenancy attribute. You can't change the instance tenancy of a VPC after you create it. This attribute has the following values.

Value	Description
`default`	An instance launched into the VPC is a Dedicated Instance if the tenancy attribute for the instance is `dedicated`.
`dedicated`	All instances launched into the VPC are Dedicated Instances, regardless of the value of the tenancy attribute for the instance.

If you are planning to use Dedicated Instances, you can implement them using either method:

- Create the VPC with the instance tenancy set to `dedicated` (all instances launched into this VPC are Dedicated Instances).
- Create the VPC with the instance tenancy set to `default`, and specify dedicated tenancy for any instances that should be Dedicated Instances when you launch them.

Dedicated Instances Limitations

Some AWS services or their features won't work with a VPC with the instance tenancy set to `dedicated`. Check the service's documentation to confirm if there are any limitations.

Some instance types cannot be launched into a VPC with the instance tenancy set to `dedicated`. For more information about supported instances types, see Amazon EC2 Dedicated Instances.

Amazon EBS with Dedicated Instances

When you launch an Amazon EBS-backed Dedicated Instance, the EBS volume doesn't run on single-tenant hardware.

Reserved Instances with Dedicated Tenancy

To guarantee that sufficient capacity will be available to launch Dedicated Instances, you can purchase Dedicated Reserved Instances. For more information about Reserved Instances, see On-Demand and Reserved Instances.

When you purchase a Dedicated Reserved Instance, you are purchasing the capacity to launch a Dedicated Instance into a VPC at a much reduced usage fee; the price break in the hourly charge applies only if you launch an instance with dedicated tenancy. However, if you purchase a Reserved Instance with a default tenancy value, you won't get a Dedicated Reserved Instance if you launch an instance with `dedicated` instance tenancy.

In addition, you can't change the tenancy of a Reserved Instance after you've purchased it.

Auto Scaling of Dedicated Instances

For information about using Auto Scaling to launch Dedicated Instances, see Auto Scaling in Amazon Virtual Private Cloud in the *Auto Scaling Developer Guide*.

Pricing for Dedicated Instances

We have a separate pricing model for Dedicated Instances. For more information, see the Amazon EC2 Dedicated Instances product page.

Working with Dedicated Instances

This section shows you how to complete the following tasks.

Topics

- Creating a VPC with an Instance Tenancy of Dedicated (p. 190)
- Launching Dedicated Instances into a VPC (p. 190)
- Displaying Tenancy Information (p. 190)

Creating a VPC with an Instance Tenancy of Dedicated

When you create a VPC, you have the option of specifying its instance tenancy. You can accept the default, or you can specify an instance tenancy of `dedicated` for your VPC. In this section, we show you how to create a VPC with an instance tenancy of `dedicated`.

To create a VPC with an instance tenancy of dedicated (VPC Wizard)

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. From the dashboard, click the **Start VPC Wizard** button.
3. Select a VPC configuration, and then click **Select**.
4. On the next page of the wizard, select **Dedicated** from the **Hardware tenancy** list.
5. Click the **Create VPC** button to create the VPC.

To create a VPC with an instance tenancy of dedicated (Create VPC dialog box)

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Your VPCs**, and then click **Create VPC**.
3. In the **Create VPC** dialog box, select **Dedicated** from the **Tenancy** drop-down list. Specify the **CIDR Block**, and then click **Yes, Create**.

Launching Dedicated Instances into a VPC

If you launch an instance into a VPC that has an instance tenancy of `dedicated`, your instance is automatically a Dedicated Instance, regardless of the tenancy of the instance. The following procedure shows you how to launch a Dedicated Instance into a VPC that has default instance tenancy.

To launch a Dedicated Instance into a VPC with default instance tenancy

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. Create a VPC, or decide to use an existing VPC with default instance tenancy.
3. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
4. Click the **Launch Instance** button.
5. On the **Choose an Amazon Machine Image (AMI)** page, choose an AMI and click **Select**.
6. On the **Choose an Instance Type** page, select the type of instance you want to launch, then click **Next: Configure Instance Details**.
7. On the **Configure Instance Details** page, select a VPC and subnet. Select **Dedicated tenancy** from the **Tenancy** list, and then click **Next: Add Storage**.
8. Continue as prompted by the wizard. When you've finished reviewing your options on the **Review Instance Launch** page, click **Launch** to choose a key pair and launch the Dedicated Instance.

Displaying Tenancy Information

To display tenancy information for your VPC

1. Open the Amazon VPC console at https://console.amazonaws.cn/vpc/.
2. In the navigation pane, click **Your VPCs**.
3. Check the instance tenancy of your VPC in the **Tenancy** column.

4. If the **Tenancy** column is not displayed, click the **Show/Hide** button, select **Tenancy** from the **Show/Hide Columns** dialog box, and then click **Close**.

To display tenancy information for your instance

1. Open the Amazon EC2 console at https://console.amazonaws.cn/ec2/.
2. In the navigation pane, click **Instances**.
3. Check the tenancy of your instance in the **Tenancy** column.
4. If the **Tenancy** column is not displayed, do one of the following:
   - Click the **Show/Hide** button, select **Tenancy** from the **Show/Hide Columns** dialog box, and then click **Close**.
   - Select the instance. The **Description** tab in the details pane displays information about the instance, including its tenancy.

API and Command Overview

You can perform the tasks described on this page using the command line or an API. For more information about the command line interfaces and a list of available APIs, see Accessing Amazon VPC (p. 6).

Set the supported tenancy options for instances that you launch into a VPC

- create-vpc (AWS CLI)
- ec2-create-vpc (Amazon EC2 CLI)
- New-EC2Vpc (AWS Tools for Windows PowerShell)

Describe the supported tenancy options for instances launched into the VPC

- describe-vpcs (AWS CLI)
- ec2-describe-vpcs (Amazon EC2 CLI)
- Get-EC2Vpc (AWS Tools for Windows PowerShell)

Set the tenancy option for an instance

- run-instances (AWS CLI)
- ec2-run-instances (Amazon EC2 CLI)
- New-EC2Instance (AWS Tools for Windows PowerShell)

Describe the tenancy value of an instance

- describe-instances (AWS CLI)
- ec2-describe-instances (Amazon EC2 CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

Describes the tenancy value of a Reserved Instance

- describe-reserved-instances (AWS CLI)
- ec2-describe-reserved-instances (Amazon EC2 CLI)

- Get-EC2ReservedInstance (AWS Tools for Windows PowerShell)

Describes the tenancy value of a Reserved Instance offering

- describe-reserved-instances-offerings (AWS CLI)
- ec2-describe-reserved-instances-offerings (Amazon EC2 CLI)
- Get-EC2ReservedInstancesOffering (AWS Tools for Windows PowerShell)

ClassicLink

ClassicLink allows you to link an EC2-Classic instance to a VPC in your account, within the same region. This allows you to associate the VPC security groups with the EC2-Classic instance, enabling communication between your EC2-Classic instance and instances in your VPC using private IP addresses. ClassicLink removes the need to make use of public IP addresses or Elastic IP addresses to enable communication between instances in these platforms. For more information about private and public IP addresses, see IP Addressing in Your VPC (p. 116).

ClassicLink is available to all users with accounts that support the EC2-Classic platform, and can be used with any EC2-Classic instance.

There is no additional charge for using ClassicLink. Standard charges for data transfer and instance hour usage apply.

For more information about ClassicLink and how to use it, see the following topics in the *Amazon EC2 User Guide*:

- ClassicLink Basics
- ClassicLink Limitations
- Working with ClassicLink
- ClassicLink API and CLI Overview

Amazon VPC Limits

The following table lists the limits related to Amazon VPC. Unless indicated otherwise, you can request an increase for any of these limits by using the Amazon VPC Limits form.

Resource	Default limit	Comments
VPCs per region	5	This limit can be increased upon request. The limit for Internet gateways per region is directly correlated to this one. Increasing this limit will increase the limit on Internet gateways per region by the same amount.
Subnets per VPC	200	This limit can be increased upon request.
Internet gateways per region	5	This limit is directly correlated with the limit on VPCs per region. You cannot increase this limit individually; the only way to increase this limit is to increase the limit on VPCs per region. Only one Internet gateway can be attached to a VPC at a time.
Virtual private gateways per region	5	This limit can be increased upon request; however, only one virtual private gateway can be attached to a VPC at a time.
Customer gateways per region	50	This limit can be increased upon request.
VPN connections per region	50	This limit can be increased upon request.
VPN connections per VPC (per virtual private gateway)	10	This limit can be increased upon request.
Route tables per VPC	200	Including the main route table. You can associate one route table to one or more subnets in a VPC.
Entries per route table	50	This is the limit for the number of non-propagated entries per route table. This limit can be increased upon request; however, network performance may be impacted.

Resource	Default limit	Comments
Elastic IP addresses per region for each AWS account	5	This is the limit for the number of VPC Elastic IPs you can allocate within a region. This is a separate limit from the EC2 Elastic IP address limit. This limit can be increased upon request.
Security groups per VPC	100	You can request an increase for this limit; however, an increase is not guaranteed. Network performance may be impacted, depending on the way the security groups are configured. We may reject a request if such a performance risk exists.
Rules per security group	50	This limit can be increased or decreased upon request, however, the multiple of the limit for rules per security group and the limit for security groups per network interface cannot exceed 250. For example, if you want 100 rules per security group, we decrease your number of security groups per network interface to 2.
Security groups per network interface	5	This limit can be increased or decreased upon request; up to a maximum of 16. The multiple of the limit for security groups per network interface and the limit for rules per security group cannot exceed 250. For example, if you want 10 security groups per network interface, we decrease your number of rules per security group to 25.
Network interfaces per instance	-	This limit varies by instance type. For more information, see Private IP Addresses Per ENI Per Instance Type.
Network interfaces per VPC	100	This limit is calculated by multiplying your On-Demand instance limit by 5. The default limit for On-Demand instances is 20. You can increase the number of network interfaces per VPC by request, or by increasing your On-Demand instance limit.
Network ACLs per VPC	200	You can associate one network ACL to one or more subnets in a VPC. This limit is not the same as the number of rules per network ACL.
Rules per network ACL	20	This is the one-way limit for a single network ACL, where the limit for ingress rules is 20, and the limit for egress rules is 20.
BGP Advertised Routes per VPN Connection	100	This limit cannot be increased. If you require more than 100 prefixes, advertise a default route.

Resource	Default limit	Comments
Active VPC peering connections per VPC	50	This limit can be increased via special request to AWS Support. The maximum limit is 125 peering connections per VPC. The number of entries per route table should be increased accordingly; however, network performance may be impacted.
Outstanding VPC peering connection requests	25	This is the limit for the number of outstanding VPC peering connection requests that you've requested from your account. This limit can be increased via special request to AWS Support.
Expiry time for an unaccepted VPC peering connection request	1 week (168 hours)	This limit can be increased via special request to AWS Support.
VPC endpoints per region	20	This limit can be increased upon request; up to a maximum of 255 endpoints per VPC.
Flow logs per single network interface, single subnet, or single VPC in a region	2	You can effectively have 6 flow logs per network interface if you create 2 flow logs for the subnet, and 2 flow logs for the VPC in which your network interface resides. This limit cannot be increased.

Document History

The following table describes the important changes in each release of this Amazon VPC guide.

Feature	API Version	Description	Release Date
VPC Flow Logs	2015-04-15	You can create a flow log to capture information about the IP traffic going to and from network interfaces in your VPC. For more information, see VPC Flow Logs (p. 106).	10 June 2015
VPC Endpoints	2015-03-01	An endpoint enables you to create a private connection between your VPC and another AWS service without requiring access over the Internet, through a VPN connection, through a NAT instance, or through AWS Direct Connect. For more information, see VPC Endpoints (p. 160).	11 May 2015
ClassicLink	2014-10-01	ClassicLink allows you to link your EC2-Classic instance to a VPC in your account. You can associate VPC security groups with the EC2-Classic instance, enabling communication between your EC2-Classic instance and instances in your VPC using private IP addresses. For more information, see ClassicLink (p. 193).	7 January 2015
Use private hosted zones	2014-09-01	You can access resources in your VPC using custom DNS domain names that you define in a private hosted zone in Amazon Route 53. For more information, see Using Private Hosted Zones (p. 152).	5 November 2014
Modify a subnet's public IP addressing attribute	2014-06-15	You can modify the public IP addressing attribute of your subnet to indicate whether instances launched into that subnet should receive a public IP address. For more information, see Modifying Your Subnet's Public IP Addressing Behavior (p. 117).	21 June 2014

Feature	API Version	Description	Release Date
VPC peering	2014-02-01	You can create a VPC peering connection between two VPCs, which allows instances in either VPC to communicate with each other using private IP addresses - as if they are within the same VPC. For more information, see VPC Peering (p. 152).	24 March 2014
New EC2 launch wizard	2013-10-01	Added information about the redesigned EC2 launch wizard. For more information, see Step 3: Launch an Instance into Your VPC (p. 13).	10 October 2013
Assigning a public IP address	2013-07-15	Added information about a new public IP addressing feature for instances launched in a VPC. For more information, see Assigning a Public IP Address During Launch (p. 118).	20 August 2013
Enabling DNS hostnames and disabling DNS resolution	2013-02-01	By default, DNS resolution is enabled. You can now disable DNS resolution using the Amazon VPC console, the Amazon EC2 command line interface, or the Amazon EC2 API actions. By default, DNS hostnames are disabled for nondefault VPCs. You can now enable DNS hostnames using the Amazon VPC console, the Amazon EC2 command line interface, or the Amazon EC2 API actions. For more information, see Using DNS with Your VPC (p. 149).	11 March 2013
VPN connections using static routing configuration	2012-08-15	You can create IPsec VPN connections to Amazon VPC using static routing configurations. Previously, VPN connections required the use of the Border Gateway Protocol (BGP). We now support both types of connections and are excited to announce that you can now establish connectivity from devices that do not support BGP, including Cisco ASA and Microsoft Windows Server 2008 R2.	13 September 2012
Automatic route propagation	2012-08-15	You can now configure automatic propagation of routes from your VPN and Direct Connect links to your VPC routing tables. This feature simplifies the effort to create and maintain connectivity to Amazon VPC.	13 September 2012
AWS VPN CloudHub and redundant VPN connections		You can securely communicate from one site to another with or without a VPC. You can use redundant VPN connections to provide a fault-tolerant connection to your VPC.	29 September 2011
VPC Everywhere	2011-07-15	Support in five AWS regions, VPCs in multiple Availability Zones, multiple VPCs per AWS account, multiple VPN connections per VPC, Microsoft Windows Server 2008 R2 and Microsoft SQL Server Reserved Instances.	03 August 2011

Feature	API Version	Description	Release Date
Dedicated Instances	2011-02-28	Dedicated Instances are Amazon EC2 instances launched within your VPC that run hardware dedicated to a single customer. Dedicated Instances let you take full advantage of the benefits of Amazon VPC and AWS elastic provisioning, pay only for what you use, and a private, isolated virtual network—all while isolating your instances at the hardware level.	27 March 2011

AWS Glossary

Blank

placeholder — This page redirects to the AWS Glossary in the *AWS General Reference*.